Deep Q-Network (DQN) on LunarLander-v2

> In this post, We will take a hands-on-lab of Simple Deep Q-Network (DQN) on openAI LunarLander-v2 environment. This is the coding exercise from udacity Deep Reinforcement Learning Nanodegree.

- toc: true
- badges: true
- comments: true
- author: Chanseok Kang
- categories: [Python, Reinforcement_Learning, PyTorch, Udacity]
- image: images/LunarLander-v2.gif

Deep Q-Network (DQN)

In this notebook, you will implement a DQN agent with OpenAI Gym's LunarLander-v2 environment.

Import the Necessary Packages

In [1]:
```
import gymnasium as gym
import random
import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import numpy as np
import time
import base64, io
import os
from datetime import datetime
from collections import deque, namedtuple
from tqdm.notebook import tqdm
from copy import deepcopy
from gymnasium.wrappers import RecordVideo
from IPython.display import HTML
from IPython import display
import glob
```

```
from tqdm.autonotebook import tqdm as notebook_tqdm
```

```
/tmp/ipykernel_59590/3197289097.py:22: TqdmExperimentalWarning: Using `
tqdm.autonotebook.tqdm` in notebook mode. Use `tqdm.tqdm` instead to fo
rce console mode (e.g. in jupyter console)
  from tqdm.autonotebook import tqdm as notebook_tqdm
```

Instantiate the Environment and Agent

Initialize the environment.

In [2]:

```
# Create environment with Gymnasium instead of Gym
env = gym.make('LunarLander-v3', render_mode=None)
# Random seed is set differently in Gymnasium
env.reset(seed=0)  # Sets seed for the environment
torch.manual_seed(0)  # Set PyTorch seed
np.random.seed(0)  # Set NumPy seed
random.seed(0)  # Set Python's random seed

print('State shape: ', env.observation_space.shape)
print('Number of actions: ', env.action_space.n)
```

```
State shape:  (8,)
Number of actions:  4
```

Define Neural Network Architecture.

Since `LunarLander-v3` environment is sort of simple envs, we don't need complicated architecture. We just need non-linear function approximator that maps from state to action.

In [3]:

```
class QNetwork(nn.Module):
    """Actor (Policy) Model."""

    def __init__(self, state_size, action_size, seed, fc1_units=64, fc
        """Initialize parameters and build model.
        Params
        ======
            state_size (int): Dimension of each state
            action_size (int): Dimension of each action
            seed (int): Random seed
            fc1_units (int): Number of nodes in first hidden layer
            fc2_units (int): Number of nodes in second hidden layer
        """
        super(QNetwork, self).__init__()
        self.seed = torch.manual_seed(seed)
        self.fc1 = nn.Linear(state_size, fc1_units)
        self.fc2 = nn.Linear(fc1_units, fc2_units)
        self.fc3 = nn.Linear(fc2_units, action_size)
```

```python
    def forward(self, state):
        """Build a network that maps state -> action values."""
        x = F.relu(self.fc1(state))
        x = F.relu(self.fc2(x))
        return self.fc3(x)
```

Define some hyperparameter

In [4]:
```python
BUFFER_SIZE = int(1e5)  # replay buffer size
BATCH_SIZE = 64         # minibatch size
GAMMA = 0.99            # discount factor
TAU = 1e-3              # for soft update of target parameters
LR = 5e-4               # learning rate
UPDATE_EVERY = 4        # how often to update the network
```

In [5]:
```python
device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu"
```

Define Replay Buffer

This class implements a fixed-size replay buffer for storing experience tuples in reinforcement learning. The idea behind the replay buffer is to hold a history of transitions (state, action, reward, next_state, done) for later random sampling during the training process. This helps break the correlation between consecutive experiences and smooths over changes in the data distribution, leading to more stable and sample-efficient learning.

Below is a breakdown of each component:

- **Constructor (`__init__`)**
  This function initializes the ReplayBuffer object with the following key components:
  - `action_size` : The dimension of the action space, which might be used if any reshaping or processing based on action dimensions is needed.
  - `buffer_size` : The maximum number of experiences (transitions) the replay buffer can store. Internally, this is managed with a Python `deque` (double-ended queue) that discards the oldest experiences once this maximum capacity is reached.
  - `batch_size` : The number of experiences to sample at once during learning, which corresponds to the minibatch size used in training the model.
  - `seed` : A value used to initialize Python's random number generator for reproducibility.

- `experience`: A named tuple with fields `["state", "action", "reward", "next_state", "done"]` that represents a single transition or experience tuple.

- **`add` Function**
  This function implements the addition of a new experience tuple to the memory buffer. When you call:

$$e = (\text{state},\ \text{action},\ \text{reward},\ \text{next\\_state},\ \text{done})$$

  the tuple is created using the predefined named tuple, and then appended to the `deque` representing the memory. If the `deque` is full, the oldest experience is automatically removed. This is analogous to a moving window over the last N experiences, with $N = \text{buffer\\_size}$.

- **`sample` Function**
  This function implements the random sampling of a batch of experiences from the memory. It works as follows:

  1. Randomly select a set of k experiences (where $k = \text{batch\\_size}$) from the stored experiences.

  2. Extract each field (state, action, reward, next_state, done) from the sampled experiences and convert them into tensors.

     Mathematically, let E be the set of all experience tuples stored in the memory. The sampling process selects a subset (E_{\text{batch}} \subset E) such that:

$$|E_{\text{batch}}| = \text{batch\\_size}$$

     For each component (say state), the function stacks the corresponding values:

$$\text{states} = \text{stack}\{e.\text{state} \mid e \in E_{\text{batch}}\}$$

     This is performed for all components, and the resulting batches are returned as a tuple.

- **`__len__` Function**
  This function returns the current number of experiences stored in the replay buffer. If (m) is the number of stored experience tuples, then:

$$\text{len}(\text{ReplayBuffer}) = m$$

  This gives an idea of how many transitions are available in the buffer for sampling.

In [6]:
```
class ReplayBuffer:
    """Fixed-size buffer to store experience tuples."""

    def __init__(self, action_size, buffer_size, batch_size, seed):
        """Initialize a ReplayBuffer object."""
        self.action_size = action_size
        self.memory = deque(maxlen=buffer_size)
        self.batch_size = batch_size
        self.experience = namedtuple("Experience", field_names=["state
        self.seed = random.seed(seed)

    def add(self, state, action, reward, next_state, done):
        """Add a new experience to memory."""
        e = self.experience(state, action, reward, next_state, done)
        self.memory.append(e)

    def sample(self):
        """Randomly sample a batch of experiences from memory."""
        experiences = random.sample(self.memory, k=self.batch_size)

        states = torch.from_numpy(np.vstack([e.state for e in experien
        actions = torch.from_numpy(np.vstack([e.action for e in experi
        rewards = torch.from_numpy(np.vstack([e.reward for e in experi
        next_states = torch.from_numpy(np.vstack([e.next_state for e i
        dones = torch.from_numpy(np.vstack([e.done for e in experience

        return (states, actions, rewards, next_states, dones)

    def __len__(self):
        """Return the current size of internal memory."""
        return len(self.memory)
```

Define a Base DQN Agent

This class implements a **base DQN agent** that serves as a foundation for other agents employing various exploration methods. It encapsulates the common functionality required for a Deep Q-Network (DQN) agent while leaving key methods, such as action selection and learning updates, to be defined by subclasses. Below is an explanation of each component along with its corresponding mathematical formulation:

Constructor (`__init__`):

- **Purpose:**
  Initializes the essential parameters and components for the DQN agent.

- **Components:**
  - **State and Action Size:**
    The dimensions of the state space and action space are stored in `self.state_size` and `self.action_size`, respectively.
  - **Random Seed:**
    The agent sets a seed for reproducibility using Python's random number generator:
    $$\text{seed} \rightarrow \text{random.seed}(\text{seed})$$
  - **Q-Networks:**
    Two neural networks are created:
    - `qnetwork_local`: the online (or policy) network that will be updated frequently.
    - `qnetwork_target`: the target network that is updated more slowly, used to stabilize learning.

    These networks estimate the action-value function:
    $$Q(s, a; \theta)$$
    where θ represents the network parameters.
  - **Optimizer:**
    An Adam optimizer is used to update the parameters of the local Q-network.
  - **Replay Memory:**
    An instance of a replay buffer is created to store experience tuples for later sampling:
    $$\text{experience} = (s, a, r, s', d)$$
  - **Hyperparameters:**
    Several hyperparameters are initialized:
    - $$\gamma = 0.99$$
      (discount factor)
    - $$\tau = 1 \times 10^{-3}$$
      (soft update parameter for target network)

- $$\text{UPDATE\_EVERY} = 4$$

  (frequency of learning updates)

- $$\text{batch\_size} = 64$$

- $$\text{learning\_start} = 1000$$

  (number of steps to wait before training begins)

- **Method Name:**
  `self.method_name` is set as `"Base DQN"` for reporting purposes.

Method `step`:

- **Purpose:**
  Manages storing experiences and triggering learning updates.

- **Functionality:**

  1. **Store Experience:**
     Each experience tuple

     $$(s, a, r, s', d)$$

     is added to the replay memory.

  2. **Periodic Learning:**
     The agent checks whether it is time to learn based on the variable `self.t_step`:

     $$\text{t\_step} = (\text{t\_step} + 1) \mod \text{UPDATE\_EVERY}$$

     After a fixed number of steps and once enough experiences are collected (i.e. total steps $>$ `learning_start` and buffer has at least `batch_size` experiences), the agent samples a batch from replay memory and calls the `learn` method with:

     $$\text{Learn}(E, \gamma)$$

     where E is the batch of experiences and γ is the discount factor.

Abstract Methods `act` and `learn`:

- **Purpose:**
  These methods are placeholders and must be implemented by subclasses to

define the action-selection policy (e.g., ϵ-greedy, Boltzmann, etc.) and the learning algorithm (updating the Q-network parameters).

- **act:**

  $$\text{action} = \pi(s)$$

  where $\pi(s)$ is the policy dictated by the chosen exploration or decision-making strategy.

- **learn:**
  Updates the network parameters using the loss computed from a batch of experience tuples. A typical DQN loss is:

  $$L(\theta) = \mathbb{E}_{(s,a,r,s',d)\sim E}\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta)\right)^2\right]$$

  where θ^- are the parameters of the target network.

Method `soft_update`:

- **Purpose:**
  Smoothly updates the target Q-network parameters towards the local Q-network parameters.

- **Update Rule:**
  For each parameter:

  $$\theta_{\text{target}} \leftarrow \tau\theta_{\text{local}} + (1-\tau)\theta_{\text{target}}$$

  This prevents abrupt changes in the target network, contributing to learning stability.

Method `save`:

- **Purpose:**
  Saves the model parameters (for both Q-networks and the optimizer state) to a file.

- **Mathematical Note:**
  Saving does not involve a mathematical operation, but conceptually it writes:

  $$\theta_{\text{local}},\ \theta_{\text{target}},\ \text{optimizer state} \quad \text{to disk}$$

Method `load` :

- **Purpose:**
  Loads the stored model parameters, allowing the agent to continue training or evaluation from a saved state.

- **Operation:**
  Checks if the file exists and then loads:

$$\theta_{\text{local}},\ \theta_{\text{target}},\ \text{optimizer state} \quad \text{from disk}$$

In [7]:
```
class DQNAgentBase:
    """Deep Q-Network Agent base class."""

    def __init__(self, state_size, action_size, seed=0):
        """Initialize a DQN Agent Base.

        Params
        ======
            state_size (int): dimension of each state
            action_size (int): dimension of each action
            seed (int): random seed
        """
        self.state_size = state_size
        self.action_size = action_size
        self.seed = seed
        random.seed(seed)

        # Q-Networks
        self.qnetwork_local = QNetwork(state_size, action_size, seed).
        self.qnetwork_target = QNetwork(state_size, action_size, seed)
        self.optimizer = optim.Adam(self.qnetwork_local.parameters(),

        # Replay memory
        self.memory = ReplayBuffer(action_size, BUFFER_SIZE, BATCH_SIZ

        # Initialize time step (for updating every UPDATE_EVERY steps)
        self.t_step = 0
        self.method_name = "Base DQN"
        self.TAU = TAU  # Soft update parameter

    def get_init_params(self):
        """Return parameters needed to initialize this agent."""
        return {
            'state_size': self.state_size,
            'action_size': self.action_size,
            'seed': self.seed
        }

    def step(self, state, action, reward, next_state, done, total_step
```

```
        """Store experience in replay memory, and use random sample to

        Params
        ======
            state: current state
            action: action taken
            reward: reward received
            next_state: next state
            done: whether episode is done
            total_steps: total steps taken across all episodes (option
        """
        # Save experience in replay memory
        self.memory.add(state, action, reward, next_state, done)

        # Learn every UPDATE_EVERY time steps
        self.t_step = (self.t_step + 1) % UPDATE_EVERY
        if self.t_step == 0:
            # If enough samples are available in memory, get random su
            if len(self.memory) > BATCH_SIZE:
                experiences = self.memory.sample()
                self.learn(experiences, GAMMA)

    def act(self, state, training=True):
        """Returns actions for given state as per current policy."""
        pass  # To be implemented by subclasses

    def learn(self, experiences, gamma):
        """Update value parameters using given batch of experience tup
        pass  # To be implemented by subclasses

    def soft_update(self, local_model, target_model, tau):
        """Soft update model parameters.
        θ_target = τ*θ_local + (1 - τ)*θ_target

        Params
        ======
            local_model: PyTorch model (weights will be copied from)
            target_model: PyTorch model (weights will be copied to)
            tau (float): interpolation parameter
        """
        for target_param, local_param in zip(target_model.parameters()
            target_param.data.copy_(tau*local_param.data + (1.0-tau)*t

    def save(self, filename):
        """Save the model parameters."""
        torch.save(self.qnetwork_local.state_dict(), filename)

    def load(self, filename):
        """Load model parameters."""
        self.qnetwork_local.load_state_dict(torch.load(filename))
        self.qnetwork_target.load_state_dict(torch.load(filename))
```

Define Training Pipeline

This function trains a DQN agent on the LunarLander-v3 environment over multiple episodes. The main idea is to let the agent interact with the environment, collect experiences, update its policy through learning steps, and track its performance over episodes.

Here's a brief rundown of the key steps:

- **Environment Creation and Initialization:**
  The function creates the LunarLander-v3 environment and initializes performance metrics like scores and a sliding window for recent scores.

- **Episode Loop:**
  For each episode, the environment is reset with a different seed, and the agent interacts with it for a fixed maximum number of timesteps. At each timestep:

  - The agent selects an action (using its exploration strategy).
  - The environment returns the next state, reward, and a done flag.
  - The agent stores the experience and possibly learns from it.
  - The total reward accumulates into a score for the episode.
- **Exploration and Logging:**
  If the agent uses epsilon-greedy exploration, epsilon is updated over time. Similarly, if the agent uses softmax policy for exploration, the temperature is updated over time. The function prints progress every few episodes and calculates the average score over recent episodes.

- **Early Stopping:**
  The training loop stops early if the average score over the latest episodes meets or exceeds a specified threshold (early_stop_score).

In [8]:
```
def train_agent(agent, env_name='LunarLander-v3', n_episodes=1000, max
    """Train a DQN agent."""
    # Create environment
    env = gym.make(env_name)

    # Initialize scores
    scores = []
    scores_window = deque(maxlen=100)
    eps = 1.0  # For epsilon-greedy agents
    start_time = time.time()
    total_steps = 0

    # Train for n_episodes
```

```
    for i_episode in tqdm(range(1, n_episodes+1)):
        state, _ = env.reset(seed=i_episode)  # Different seed each ep
        score = 0

        for t in range(max_t):
            # Select action
            action = agent.act(state, training=True)

            # Take action
            next_state, reward, terminated, truncated, _ = env.step(ac
            done = terminated or truncated

            # Increment total steps counter
            total_steps += 1

            # Learn from experience
            agent.step(state, action, reward, next_state, done, total_

            # Move to the next state
            state = next_state
            score += reward

            if done:
                break

        # Update epsilon if the agent uses epsilon-greedy exploration
        if hasattr(agent, 'update_epsilon'):
            agent.update_epsilon()

        if hasattr(agent, 'update_temperature'):
            agent.update_temperature()

        # Save score and check if environment is solved
        scores_window.append(score)
        scores.append(score)

        # Print progress
        if i_episode % print_every == 0:
            end_time = time.time()
            elapsed_time = end_time - start_time
            avg_score = np.mean(scores_window)
            print(f"\rEpisode {i_episode}/{n_episodes} | Avg Score: {a
            if hasattr(agent, 'eps'):
                print(f"Epsilon: {agent.eps:.4f}")
            start_time = end_time

        # Check if the environment is solved
        if np.mean(scores_window) >= early_stop_score:
            print(f"\nEnvironment solved in {i_episode} episodes!\tAve
            break

    return scores
```

Evaluate the trained DQN Agents

- `evaluate_agent`
  Runs the learned agent for a fixed number of episodes and returns the average and standard deviation of the scores. It simply resets the environment, collects rewards until the episode ends, and aggregates these scores across episodes.

- `benchmark_agents`
  Trains and evaluates a list of agents. For each agent, it runs multiple trials (each with a fresh agent copy), recording training scores, evaluation scores, and training time. It then aggregates these results into a DataFrame and also keeps all training score curves for later plotting. This function enables comparing different agents side-by-side in terms of their performance, convergence speed, and overall training cost.

- `plot_training_curves`
  Takes the stored training score curves from different agents (averaged across multiple trials) and plots them. It smooths the curves and includes shaded areas representing the standard deviation so that you can visually compare how the agents' performance improves over episodes.

- `plot_benchmark_results`
  Uses the results DataFrame to create various bar charts that compare the agents' evaluation performance, the number of episodes required to converge, training time, and final training scores. This helps summarize the performance metrics across different agents in a visual format.

In [9]:
```
# Evaluation function
def evaluate_agent(agent, env_name='LunarLander-v3', n_episodes=20):
    """Evaluate a trained agent's performance."""
    env = gym.make(env_name, render_mode=None)
    scores = []

    for i in range(n_episodes):
        state, _ = env.reset(seed=i+1000)  # Different seeds from trai
        score = 0
        done = False

        while not done:
            action = agent.act(state, training=False)  # No exploratio
            next_state, reward, terminated, truncated, _ = env.step(ac
            done = terminated or truncated
            state = next_state
            score += reward
```

```
        scores.append(score)

    return np.mean(scores), np.std(scores)
```

In [10]:
```
def benchmark_agents(agents, env_name='LunarLander-v3', n_episodes=100
    """Train and evaluate multiple agents to compare their performance
    results = {
        'agent_name': [],
        'training_episodes': [],
        'final_avg_score': [],
        'eval_avg_score': [],
        'eval_std_score': [],
        'training_time': []
    }

    all_training_scores = {}  # To store training curves for each agen

    for agent in agents:
        print(f"\n\n{'-'*50}")
        print(f"Training agent: {agent.method_name}")
        print(f"{'-'*50}")

        # Train for n_trials and collect results
        trial_episodes = []
        trial_final_scores = []
        trial_eval_scores = []
        trial_eval_stds = []
        trial_times = []
        trial_training_scores = []

        for trial in range(n_trials):
            print(f"\nTrial {trial+1}/{n_trials}")

            # Create a fresh copy of the agent for each trial
            if trial > 0:
                # Recreate agent of the same type with new seed
                agent_class = agent.__class__

                # Get initialization parameters using the get_init_par
                agent_args = agent.get_init_params()

                # Handle seed properly - it might be None or already u
                new_seed = (agent_args.get('seed', 0) or 0) + 100 * tr
                agent_args['seed'] = new_seed

                # Create new agent instance
                agent = agent_class(**agent_args)

            # Train the agent
            start_time = time.time()
            scores = train_agent(agent, env_name=env_name, n_episodes=
```

```
            end_time = time.time()
            training_time = end_time - start_time

            # Evaluate the agent
            eval_score, eval_std = evaluate_agent(agent, env_name=env_

            # Save model if needed
            if False:  # Set to True if you want to save models
                os.makedirs('models', exist_ok=True)
                filename = f"models/{agent.method_name.replace(' ', '_
                agent.save(filename)
                print(f"Model saved to {filename}")

            # Record results
            trial_episodes.append(len(scores))
            trial_final_scores.append(np.mean(scores[-100:]))
            trial_eval_scores.append(eval_score)
            trial_eval_stds.append(eval_std)
            trial_times.append(training_time)
            trial_training_scores.append(scores)

        # Average results across trials
        results['agent_name'].append(agent.method_name)
        results['training_episodes'].append(np.mean(trial_episodes))
        results['final_avg_score'].append(np.mean(trial_final_scores))
        results['eval_avg_score'].append(np.mean(trial_eval_scores))
        results['eval_std_score'].append(np.mean(trial_eval_stds))
        results['training_time'].append(np.mean(trial_times))

        # Store all training scores for plotting
        all_training_scores[agent.method_name] = trial_training_scores

    # Convert to DataFrame
    results_df = pd.DataFrame(results)
    return results_df, all_training_scores


def plot_training_curves(all_training_scores, title="Training Curves")
    """Plot training curves for multiple agents."""
    plt.figure(figsize=(12, 8))

    for agent_name, trial_scores in all_training_scores.items():
        # Average scores across trials
        # First, find the shortest trial length
        min_length = min(len(scores) for scores in trial_scores)

        # Truncate all trials to the same length
        truncated_scores = [scores[:min_length] for scores in trial_sc

        # Calculate average and standard deviation
        avg_scores = np.mean(truncated_scores, axis=0)
        std_scores = np.std(truncated_scores, axis=0)
```

```
        # Create x-axis
        episodes = np.arange(1, min_length + 1)

        # Smooth the curves for better visualization
        window_size = min(100, min_length // 10)
        smoothed_scores = np.convolve(avg_scores, np.ones(window_size)
        smoothed_episodes = episodes[window_size-1:]

        # Plot with shaded standard deviation area
        plt.plot(smoothed_episodes, smoothed_scores, label=agent_name)
        plt.fill_between(
            smoothed_episodes,
            smoothed_scores - std_scores[window_size-1:],
            smoothed_scores + std_scores[window_size-1:],
            alpha=0.2
        )

    plt.xlabel('Episode')
    plt.ylabel('Score')
    plt.title(title)
    plt.legend()
    plt.grid(True)
    plt.show()

def plot_benchmark_results(results_df):
    """Plot comparison of different agents."""
    # Bar chart for evaluation scores
    plt.figure(figsize=(14, 10))

    # Create a plot with standard deviation error bars
    plt.subplot(2, 2, 1)
    scores = results_df['eval_avg_score']
    stds = results_df['eval_std_score']

    bars = plt.bar(results_df['agent_name'], scores, yerr=stds, capsiz
    plt.title('Evaluation Performance')
    plt.ylabel('Average Score')
    plt.xlabel('')
    plt.xticks(rotation=45, ha='right')

    # Add values on top of the bars
    for i, bar in enumerate(bars):
        height = bar.get_height()
        plt.text(bar.get_x() + bar.get_width()/2., height + stds[i],
                f'{scores[i]:.1f}', ha='center', va='bottom')

    # Plot training episode counts (lower is better)
    plt.subplot(2, 2, 2)
    bars = plt.bar(results_df['agent_name'], results_df['training_epis
    plt.title('Episodes until Convergence')
    plt.ylabel('Episodes')
```

```
    plt.xlabel('')
    plt.xticks(rotation=45, ha='right')

    # Add values on top of the bars
    for bar in bars:
        height = bar.get_height()
        plt.text(bar.get_x() + bar.get_width()/2., height,
                f'{height:.0f}', ha='center', va='bottom')

    # Plot training time
    plt.subplot(2, 2, 3)
    bars = plt.bar(results_df['agent_name'], results_df['training_time
    plt.title('Training Time')
    plt.ylabel('Time (minutes)')
    plt.xlabel('')
    plt.xticks(rotation=45, ha='right')

    # Add values on top of the bars
    for bar in bars:
        height = bar.get_height()
        plt.text(bar.get_x() + bar.get_width()/2., height,
                f'{height:.1f}', ha='center', va='bottom')

    # Plot final training scores
    plt.subplot(2, 2, 4)
    bars = plt.bar(results_df['agent_name'], results_df['final_avg_sco
    plt.title('Final Training Score')
    plt.ylabel('Average Score')
    plt.xlabel('')
    plt.xticks(rotation=45, ha='right')

    # Add values on top of the bars
    for bar in bars:
        height = bar.get_height()
        plt.text(bar.get_x() + bar.get_width()/2., height,
                f'{height:.1f}', ha='center', va='bottom')

    plt.tight_layout()
    plt.show()
```

Epsilon Greedy

DQN_EpsilonGreedy

This class implements a DQN agent that uses the standard ϵ-greedy exploration strategy. It inherits from a base DQN class, and adds parameters and methods to manage the decaying ϵ value. The agent learns by updating its Q-values using temporal difference (TD) learning with a fixed target network.

(a) **Exploration Strategy: ϵ-greedy**

- **Action Selection:**
The method `act` is defined as follows:

 1. **Evaluate the Q-network:**
 The agent passes the current state s through its **local Q-network** $Q(s, \cdot; \theta)$ (where θ denotes the learned parameters) to get the corresponding action-values:

 $$Q(s, a; \theta) \quad \text{for every } a \in \{0, 1, \ldots, A-1\}.$$

 2. **Random Action vs. Greedy Action:**
 With probability ϵ, the agent selects a random action:

 $$\text{action} \sim \text{Uniform}\{0, 1, \ldots, A-1\},$$

 and with probability $(1-\epsilon)$ the agent selects the action with the highest Q-value:

 $$\text{action} = \arg\max_a \, Q(s, a; \theta).$$

- **Epsilon Decay:**
The method `update_epsilon` implements a decay schedule:

$$\epsilon \leftarrow \max(\epsilon_{\text{end}}, \, \epsilon_{\text{decay}} \times \epsilon)$$

where:

 - ϵ_{start} is the initial value,
 - ϵ_{end} is the minimum value, and
 - ϵ_{decay} is the factor by which ϵ is multiplied at each update.
 This ensures that initially the agent explores more (high ϵ) and gradually shifts toward a greedy policy.

(b) **Q-Network Training: Temporal-Difference (TD) Learning**

- **Sampling and Updates (learn method):**
The agent obtains a batch of experiences (tuples) (s, a, r, s', d), where d indicates if the next state is terminal. The training update for each experience follows the TD target:

 1. **Target Q-value Calculation:**
 Using the **target Q-network** $Q(s', \cdot; \theta^-)$ (with fixed parameters θ^-), the

maximum Q-value is selected for the next state s':

$$Q_{\text{target}}(s') = \max_{a'} Q(s', a'; \theta^{-}).$$

The TD target for the current state is then computed as:

$$y = r + \gamma\, Q_{\text{target}}(s') \cdot (1 - d),$$

where γ is the discount factor. Note that if $d = 1$ (terminal state), the second term vanishes.

2. **Local Q-value Estimation:**
   The **local Q-network** computes $Q(s, a; \theta)$. However, since the network outputs a value for every action, the agent selects the Q-value corresponding to the executed action a:

$$Q_{\text{expected}} = Q(s, a; \theta).$$

3. **Loss and Backpropagation:**
   The loss function is given by the mean squared error (MSE) between the target and expected Q-values:

$$L(\theta) = \mathbb{E}\left[\left(y - Q(s, a; \theta) \right)^2 \right].$$

   This loss is minimized via gradient descent (using an Adam optimizer). After computing the loss, the local Q-network's parameters θ are updated, and a soft update is performed on the target network:

$$\theta^{-} \leftarrow \tau\, \theta + (1 - \tau)\, \theta^{-},$$

   where τ is a small constant ensuring gradual updates.

---

2. DQN_FixedEpsilon

This class is a simple variant of the above, where the exploration parameter ϵ is kept fixed. It inherits from `DQN_EpsilonGreedy` but overrides the ϵ update method to do nothing; that is, ϵ remains constant throughout training.

(a) Exploration Strategy: Fixed ϵ

- **Action Selection:**
  The `act` method is exactly the same as in `DQN_EpsilonGreedy`:

- With probability ϵ (a fixed value), a random action is chosen.
- Otherwise, the greedy action based on the Q-values is selected.

- **Fixed Exploration Parameter:**
The method `update_epsilon` is overridden such that it returns ϵ without any modifications:

$$\epsilon \text{ remains constant.}$$

This means the agent always explores with the same probability, without any decay.

(b) **Q-Network Training:**

- **Training Process:**
The training (via `learn`) in `DQN_FixedEpsilon` is inherited from `DQN_EpsilonGreedy` and remains unchanged. That is, the same TD update procedure is followed:

  - Sample a batch of experiences.
  - Compute the TD target:

$$y = r + \gamma \max_{a'} Q(s', a'; \theta^-) \cdot (1 - d).$$

  - Evaluate the local Q-network to obtain:

$$Q(s, a; \theta).$$

  - Calculate the loss:

$$L(\theta) = \mathbb{E}\left[(y - Q(s, a; \theta))^2\right].$$

  - Perform backpropagation and update θ, followed by a soft update of θ^- using:

$$\theta^- \leftarrow \tau\,\theta + (1 - \tau)\,\theta^-.$$

Thus, the only difference between these two classes is in the handling of the exploration parameter: one decays ϵ over time, while the other keeps it constant. Both agents use the same learning process for updating the Q-network.

In [11]:
```
class DQN_EpsilonGreedy(DQNAgentBase):
    """DQN Agent with standard epsilon greedy exploration strategy."""

    def __init__(self, state_size, action_size, seed=0, eps_start=1.0,
        """Initialize a DQN Agent with Epsilon Greedy exploration."""
```

```
        super(DQN_EpsilonGreedy, self).__init__(state_size, action_siz
        self.eps_start = eps_start
        self.eps_end = eps_end
        self.eps_decay = eps_decay
        self.eps = eps_start
        self.learning_start = learning_start
        self.method_name = "Epsilon Greedy (Decaying)"
        self.t_step = 0  # Add step counter for tracking learning prog

    def get_init_params(self):
        """Return parameters needed to initialize this agent."""
        params = super().get_init_params()
        params.update({
            'eps_start': self.eps_start,
            'eps_end': self.eps_end,
            'eps_decay': self.eps_decay,
            'learning_start': self.learning_start
        })
        return params

    def act(self, state, training=True):
        """Returns actions for given state as per current policy."""
        state = torch.from_numpy(state).float().unsqueeze(0).to(device
        self.qnetwork_local.eval()
        with torch.no_grad():
            action_values = self.qnetwork_local(state)
        self.qnetwork_local.train()
        # Epsilon-greedy action selection
        if training and random.random() < self.eps:
            return random.choice(np.arange(self.action_size))
        else:
            return np.argmax(action_values.cpu().data.numpy())

    def learn(self, experiences, gamma):
        """Update value parameters using given batch of experience tup
        states, actions, rewards, next_states, dones = experiences
        # Get max predicted Q values (for next states) from target mod
        Q_targets_next = self.qnetwork_target(next_states).detach().ma
        # Compute Q targets for current states
        Q_targets = rewards + (gamma * Q_targets_next * (1 - dones))
        # Get expected Q values from local model
        Q_expected = self.qnetwork_local(states).gather(1, actions)
        # Compute loss
        loss = F.mse_loss(Q_expected, Q_targets)
        # Minimize the loss
        self.optimizer.zero_grad()
        loss.backward()
        self.optimizer.step()
        # Update target network
        self.soft_update(self.qnetwork_local, self.qnetwork_target, se

        # Increment step counter and update epsilon
```

```
        self.t_step += 1
        if self.t_step > self.learning_start:
            self.update_epsilon()

    def update_epsilon(self):
        """Update epsilon according to decay schedule."""
        self.eps = max(self.eps_end, self.eps_decay * self.eps)
        return self.eps

class DQN_FixedEpsilon(DQN_EpsilonGreedy):
    """DQN Agent with fixed epsilon exploration."""

    def __init__(self, state_size, action_size, seed=0, epsilon=0.1, l
        """Initialize a DQN Agent with fixed Epsilon Greedy exploratio
        # Initialize the parent class with proper parameters
        super(DQN_FixedEpsilon, self).__init__(
            state_size=state_size,
            action_size=action_size,
            seed=seed,
            eps_start=epsilon,  # Set initial epsilon to the fixed val
            eps_end=epsilon,    # Set min epsilon to the same fixed va
            eps_decay=1.0,      # No decay (multiply by 1.0)
            learning_start=learning_start
        )
        self.method_name = f"Fixed Epsilon ({epsilon})"

    def get_init_params(self):
        """Return parameters needed to initialize this agent."""
        # Get base parameters from DQNAgentBase (not from immediate pa
        params = super(DQN_EpsilonGreedy, self).get_init_params()
        params.update({
            'epsilon': self.eps,  # Just need the single epsilon value
            'learning_start': self.learning_start
        })
        return params

    def update_epsilon(self):
        """Epsilon remains fixed."""
        return self.eps
```

Softmax Policy (Boltzmann Exploration / Soft Q-learning)

This class implements a DQN agent that uses softmax exploration and a soft Bellman update, sometimes known as soft Q-learning.

Action Selection

In the `act` method, the current state is passed through the local Q-network to produce the Q-values for each action:

$$Q(s, a; \theta),$$

where θ represents the network parameters.

Then, if the agent is in training mode, it converts these Q-values into a probability distribution using the softmax function with a temperature parameter β (here represented as `self.temperature`):

$$\pi(a|s) = \frac{\exp\left(Q(s,a)/\beta\right)}{\sum_{a'} \exp\left(Q(s,a')/\beta\right)}.$$

A random action is then sampled according to this probability distribution. In evaluation mode, the agent simply selects the action with the highest Q-value (greedy action):

$$a^* = \arg\max_a Q(s, a; \theta).$$

Key Point:

- A **high temperature** (large β) produces a softer distribution that is closer to uniform, encouraging exploration.
- A **low temperature** (small β) makes the distribution peakier, leading to more exploitation.

Target Q-value Computation and the Soft Bellman Equation

In the `learn` method, the agent updates its Q-network using a modified version of the Bellman equation that incorporates the temperature parameter. Instead of using the standard Bellman target:

$$y = r + \gamma \max_{a'} Q(s', a'; \theta^-),$$

the agent computes a *soft* target based on a temperature-scaled LogSumExp operator. This soft Bellman target is given by:

$$y_{\text{soft}} = r + \gamma\, \beta\, \log\left(\sum_{a'} \exp\left(\frac{Q(s', a'; \theta^-)}{\beta}\right)\right),$$

where:

- r is the immediate reward,
- γ is the discount factor,
- β (here `self.temperature`) controls the softness of the max operator, and
- θ^- are the parameters of the target Q-network.

Numerical Stability in LogSumExp

The implementation uses a numerically stable version of the LogSumExp trick:

1. **Compute Maximum Q-value:**
   For stability, the maximum Q-value over actions for the next state is computed:
   $$M = \max_{a'} Q(s', a'; \theta^-).$$
2. **Scale Q-values:**
   The next Q-values are scaled by the temperature:
   $$\tilde{Q}(s', a') = \frac{Q(s', a'; \theta^-)}{\beta}.$$
3. **Compute the LogSumExp:**
   With stabilization:
   $$\log \sum_{a'} \exp\left(\tilde{Q}(s', a')/\beta - \tilde{M}\right) + \tilde{M},$$
   where $\tilde{M} = M/\beta$.
4. **Multiply Back by Temperature:**
   Finally, the output is multiplied by the temperature to obtain the soft Q value for the next state:
   $$Q_{\text{soft}}(s') = \beta \log\left(\sum_{a'} \exp\left(\frac{Q(s', a'; \theta^-)}{\beta}\right)\right).$$

The target for the current state is then computed as:

$$Q_{\text{target}} = r + \gamma\, Q_{\text{soft}}(s') \cdot (1 - d),$$

where d is a binary indicator for terminal state (so that no future reward is added when $d = 1$).

Training Overview

1. **Forward Pass:**
   For each batch, the agent obtains:

   - **Current Q-values:** $Q(s, a; \theta)$ for the taken actions (using the local Q-network).
   - **Next Q-values:** $Q(s', a'; \theta^-)$ for the next states (using the target Q-network).
2. **Loss Computation:**
   The Mean Squared Error (MSE) loss is computed between the Q-values predicted by the local network and the soft targets:

$$L(\theta) = \mathbb{E}\left[\left(Q(s, a; \theta) - \left(r + \gamma\, Q_{\text{soft}}(s') \cdot (1 - d)\right)\right)^2\right].$$

3. **Backpropagation and Optimization:**
   The loss is then backpropagated, and the optimizer updates the parameters of the local Q-network.

4. **Target Network Update:**
   A soft update is applied to the target network parameters:

$$\theta^- \leftarrow \tau\theta + (1 - \tau)\theta^-,$$

   ensuring that the target network slowly tracks the local network.

5. **Temperature Update:**
   The step counter `t_step` is incremented, and after a certain threshold (i.e., once learning has started), the temperature is decayed:

$$\beta \leftarrow \max\left(\beta_{\text{min}},\ \beta \times \text{temperature\_decay}\right).$$

Key Point:
By decaying the temperature, the agent starts with a high level of exploration and gradually shifts to more deterministic (greedy) choices as training proceeds.

Softmax DQN

You need to complete the `learn` function starting with `log_sum_exp` using the stable verion.

Softmax DQN

In [12]:
```
class DQN_Softmax(DQNAgentBase):
    """DQN Agent with softmax exploration policy."""

    def __init__(self, state_size, action_size, seed=0,
                 initial_temperature=10.0,
                 min_temperature=0.1,
                 temperature_decay=0.995,
                 learning_start=1000):
        """Initialize a DQN Agent with Softmax exploration."""
        super(DQN_Softmax, self).__init__(state_size, action_size, see
        self.temperature = initial_temperature  # Initial temperature
        self.min_temperature = min_temperature  # Minimum temperature
        self.temperature_decay = temperature_decay  # Decay rate
        self.learning_start = learning_start
        self.method_name = f"Softmax (initial_temp={initial_temperatur
        self.t_step = 0  # Step counter

    def get_init_params(self):
        """Return parameters needed to initialize this agent."""
        params = super().get_init_params()
        params.update({
            'initial_temperature': self.temperature,  # Use current te
            'min_temperature': self.min_temperature,
            'temperature_decay': self.temperature_decay,
            'learning_start': self.learning_start
        })
        return params

    def act(self, state, training=True):
        """Returns actions using softmax probabilities."""
        state = torch.from_numpy(state).float().unsqueeze(0).to(device
        self.qnetwork_local.eval()
        with torch.no_grad():
            action_values = self.qnetwork_local(state)
        self.qnetwork_local.train()

        if training:
            # Apply softmax with temperature to get action probabiliti
            probs = F.softmax(action_values / self.temperature, dim=1)
            return np.random.choice(np.arange(self.action_size), p=pro
        else:
            return np.argmax(action_values.cpu().data.numpy())

    def update_temperature(self):
        """Decrease temperature over time to reduce exploration."""
        # Only start decaying after learning has begun
        if self.t_step > self.learning_start:
            self.temperature = max(self.min_temperature,
                                   self.temperature * self.temperature_
```

```
    def learn(self, experiences, gamma):
        """Update value parameters using temperature-scaled LogSumExp
        states, actions, rewards, next_states, dones = experiences

        # Get Q values for next states from target model
        next_q_values = self.qnetwork_target(next_states)

        # Compute temperature-scaled LogSumExp of next Q values
        # For soft Q-learning: β * log(∑ exp(Q(s',a')/β))
        # Where β is the temperature parameter
        next_q_max = next_q_values.max(1)[0].unsqueeze(1)  # For numer
        scaled_next_q = next_q_values / self.temperature
        scaled_next_q_max = next_q_max / self.temperature

        # Compute log_sum_exp with temperature scaling
        log_sum_exp = scaled_next_q.sub(scaled_next_q_max).exp().sum(1

        # Multiply by temperature to get: β * log(∑ exp(Q(s',a')/β))
        soft_q_next = self.temperature * log_sum_exp

        # Compute Q targets for current states
        Q_targets = rewards + (gamma * soft_q_next * (1 - dones))

        # Get expected Q values from local model
        Q_expected = self.qnetwork_local(states).gather(1, actions)

        # Compute loss
        loss = F.mse_loss(Q_expected, Q_targets)

        # Minimize the loss
        self.optimizer.zero_grad()
        loss.backward()
        self.optimizer.step()

        # Update target network
        self.soft_update(self.qnetwork_local, self.qnetwork_target, se

        # Increment step counter and update temperature
        self.t_step += 1
        self.update_temperature()
```

Random Network Distillation

You need to complete `compute_intrinsic_reward` and `learn`.

Random Network Distillation (RND)

RNDNetwork

This helper class represents the neural networks used in the RND module. Both the **target** network and the **predictor** network share the same architecture:

1. **Architecture:**
   The network takes a state vector as input and processes it as follows:

   - A fully connected layer (`fc1`) maps the input (of dimension n) to an intermediate representation of size 64.
   - A ReLU activation is applied:

   $$h = \mathrm{ReLU}(W_1 s + b_1)$$

   - A second fully connected layer (`fc2`) maps h to an output feature vector of dimension d, where d is given by `output_size` (default is 64):

   $$\phi(s) = W_2 h + b_2.$$

2. **Purpose:**

   - **Target Network:** The target network is randomly initialized and then **frozen** (its parameters are not updated during training). It provides fixed feature representations:

   $$\phi_{\text{target}}(s).$$

   - **Predictor Network:** The predictor network is trainable and is trained to **approximate** the output of the target network given the same input state:

   $$\phi_{\text{predictor}}(s).$$

DQN_RND Class

The `DQN_RND` class extends the base DQN agent with an intrinsic motivation mechanism based on Random Network Distillation.

(a) Intrinsic Reward Computation

Key Idea:
The intrinsic reward is designed to quantify the novelty of a state. It is computed

as the prediction error between the predictor network and the fixed target network. For a given next state s', compute:

1. **Feature Extraction:**
   - **Target features:**

$$\phi_{\text{target}}(s') = f_{\text{target}}(s')$$

   - **Predictor features:**

$$\phi_{\text{predictor}}(s') = f_{\text{predictor}}(s').$$

2. **Prediction Error (Intrinsic Reward):**
   The intrinsic reward is defined as the squared error between these two vectors. Mathematically, if we use the squared Euclidean norm:

$$r_{\text{int}}(s') = \|\phi_{\text{predictor}}(s') - \phi_{\text{target}}(s')\|^2.$$

   In the code, this is computed using the mean squared error (MSE) loss with `reduction='none'` followed by a sum over the feature dimensions. This error serves as a proxy for novelty—states that are unfamiliar result in a high prediction error, thus giving a higher intrinsic reward.

(b) Combining Extrinsic and Intrinsic Rewards

When a new experience is stored, the agent computes the intrinsic reward for the next state and combines it with the extrinsic (environment) reward:

$$r_{\text{combined}} = r_{\text{ext}} + \lambda\, r_{\text{int}},$$

where λ (represented by `intrinsic_weight`) scales the contribution of the intrinsic reward relative to the extrinsic reward.

(c) Action Selection

The `act` method in this agent uses a standard **epsilon-greedy** strategy. It proceeds as follows:

- The state is passed through the local Q-network to produce $Q(s, a)$ for all actions.
- With a fixed probability (here hard-coded as 0.1 when training), a random action is chosen:

$$\text{action} \sim \text{Uniform}\{0, 1, \ldots, A-1\}.$$

- Otherwise, the agent chooses:

$$a^* = \arg\max_a Q(s, a).$$

During evaluation (when `training=False`), the agent always chooses the greedy action.

(d) Learning and Network Updates

1. **Standard Q-Network Update:**
   For a batch of experiences (s, a, r, s', d), the DQN update is computed as:

   - Compute the target Q-values using the target network:

   $$Q_{\text{target}}(s') = \max_{a'} Q(s', a'; \theta^-),$$

   and then the target value:

   $$y = r + \gamma\, Q_{\text{target}}(s') \cdot (1 - d).$$

   - The loss is the MSE between the predicted Q-value $Q(s, a; \theta)$ (obtained by gathering the Q-values corresponding to the taken actions) and y:

   $$\mathcal{L}_{\text{Q}} = \mathbb{E}\left[\big(y - Q(s, a; \theta)\big)^2\right].$$

2. **RND Predictor Update:**
   In parallel, the RND predictor network is trained to reduce the prediction error for the next states:

   $$\mathcal{L}_{\text{RND}} = \mathbb{E}\left[\left\|\phi_{\text{predictor}}(s') - \phi_{\text{target}}(s')\right\|^2\right].$$

3. **Combined Loss and Optimization:**
   The total loss is the sum of the Q-network loss and the RND predictor loss:

   $$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{Q}} + \mathcal{L}_{\text{RND}}.$$

   The optimizer then updates the parameters of both the local Q-network and the RND predictor network using backpropagation.

4. **Target Network Soft Update:**
   The target network parameters are updated using a soft update rule:

   $$\theta^- \leftarrow \tau\, \theta + (1 - \tau)\, \theta^-,$$

   where τ is a small constant ensuring a smooth update.

In [13]:
```
class RNDNetwork(nn.Module):
    """Random Network Distillation target and predictor networks"""

    def __init__(self, state_size, output_size=64, seed=0):
        super(RNDNetwork, self).__init__()
        self.seed = torch.manual_seed(seed)
        self.fc1 = nn.Linear(state_size, 64)
        self.fc2 = nn.Linear(64, output_size)

    def forward(self, state):
        x = F.relu(self.fc1(state))
        return self.fc2(x)

class DQN_RND(DQNAgentBase):
    """DQN Agent with Random Network Distillation for intrinsic motiva

    def __init__(self, state_size, action_size, seed=0, intrinsic_weig
        """Initialize RND DQN Agent."""
        super(DQN_RND, self).__init__(state_size, action_size, seed)
        self.intrinsic_weight = intrinsic_weight
        self.rnd_output_size = rnd_output_size
        self.learning_start = learning_start
        self.method_name = f"Random Network Distillation (w={intrinsic

        # RND networks - target is fixed, predictor is trained
        self.rnd_target = RNDNetwork(state_size, rnd_output_size, seed
        self.rnd_predictor = RNDNetwork(state_size, rnd_output_size, s

        # Freeze target network
        for param in self.rnd_target.parameters():
            param.requires_grad = False

        # Combine optimizers
        self.optimizer = optim.Adam(
            list(self.qnetwork_local.parameters()) + list(self.rnd_pre
            lr=5e-4
        )

    def get_init_params(self):
        """Return parameters needed to initialize this agent."""
        params = super().get_init_params()
        params.update({
            'intrinsic_weight': self.intrinsic_weight,
            'rnd_output_size': self.rnd_output_size,
            'learning_start': self.learning_start
        })
        return params

    def compute_intrinsic_reward(self, next_state):
        """Compute intrinsic reward based on prediction error."""
        next_state_tensor = torch.from_numpy(next_state).float().unsqu
```

```
        with torch.no_grad():
            target_feature = self.rnd_target(next_state_tensor)
        predictor_feature = self.rnd_predictor(next_state_tensor)

        # Intrinsic reward is the prediction error
        intrinsic_reward = F.mse_loss(predictor_feature, target_featur
        return intrinsic_reward.item()

    def act(self, state, training=True):
        """Returns actions for given state using epsilon-greedy."""
        state = torch.from_numpy(state).float().unsqueeze(0).to(device
        self.qnetwork_local.eval()
        with torch.no_grad():
            action_values = self.qnetwork_local(state)
        self.qnetwork_local.train()

        # Epsilon-greedy action selection (using fixed epsilon of 0.1
        if training and random.random() < 0.1:
            return random.choice(np.arange(self.action_size))
        else:
            return np.argmax(action_values.cpu().data.numpy())

    def step(self, state, action, reward, next_state, done, total_step
        """Save experience in replay memory with both extrinsic and in
        # Compute intrinsic reward
        intrinsic_reward = self.compute_intrinsic_reward(next_state)

        # Combine rewards
        combined_reward = reward + self.intrinsic_weight * intrinsic_r

        # Add experience to memory
        self.memory.add(state, action, combined_reward, next_state, do

        # Learn every UPDATE_EVERY time steps after learning_start ste
        self.t_step = (self.t_step + 1) % UPDATE_EVERY

        # Only use total_steps if provided, otherwise use self.t_step
        steps_taken = total_steps if total_steps is not None else self

        if self.t_step == 0 and steps_taken > self.learning_start:
            if len(self.memory) > BATCH_SIZE:
                experiences = self.memory.sample()
                self.learn(experiences, GAMMA)

    def learn(self, experiences, gamma):
        """Update value parameters and RND predictor network."""
        states, actions, rewards, next_states, dones = experiences

        # Update Q-Network (standard DQN update)
        # Get max predicted Q values (for next states) from target mod
        Q_targets_next = self.qnetwork_target(next_states).detach().ma
```

```
        # Compute Q targets for current states
        Q_targets = rewards + (gamma * Q_targets_next * (1 - dones))

        # Get expected Q values from local model
        Q_expected = self.qnetwork_local(states).gather(1, actions)

        # Compute Q-loss
        q_loss = F.mse_loss(Q_expected, Q_targets)

        # Update RND predictor network
        target_features = self.rnd_target(next_states).detach()
        predictor_features = self.rnd_predictor(next_states)
        rnd_loss = F.mse_loss(predictor_features, target_features)

        # Combined loss
        total_loss = q_loss + rnd_loss

        # Minimize the loss
        self.optimizer.zero_grad()
        total_loss.backward()
        self.optimizer.step()

        # Update target network
        self.soft_update(self.qnetwork_local, self.qnetwork_target, se

    def save(self, filename):
        """Save model including RND networks."""
        torch.save({
            'qnetwork_local_state_dict': self.qnetwork_local.state_dic
            'qnetwork_target_state_dict': self.qnetwork_target.state_d
            'rnd_target_state_dict': self.rnd_target.state_dict(),
            'rnd_predictor_state_dict': self.rnd_predictor.state_dict(
            'optimizer_state_dict': self.optimizer.state_dict(),
            'intrinsic_weight': self.intrinsic_weight
        }, filename)

    def load(self, filename):
        """Load model including RND networks."""
        if os.path.isfile(filename):
            checkpoint = torch.load(filename)
            self.qnetwork_local.load_state_dict(checkpoint['qnetwork_l
            self.qnetwork_target.load_state_dict(checkpoint['qnetwork_
            self.rnd_target.load_state_dict(checkpoint['rnd_target_sta
            self.rnd_predictor.load_state_dict(checkpoint['rnd_predict
            self.optimizer.load_state_dict(checkpoint['optimizer_state
            self.intrinsic_weight = checkpoint.get('intrinsic_weight',
            print(f"Loaded RND model from {filename}")
        else:
            print(f"No model found at {filename}")
```

The following code enables you to test each individual

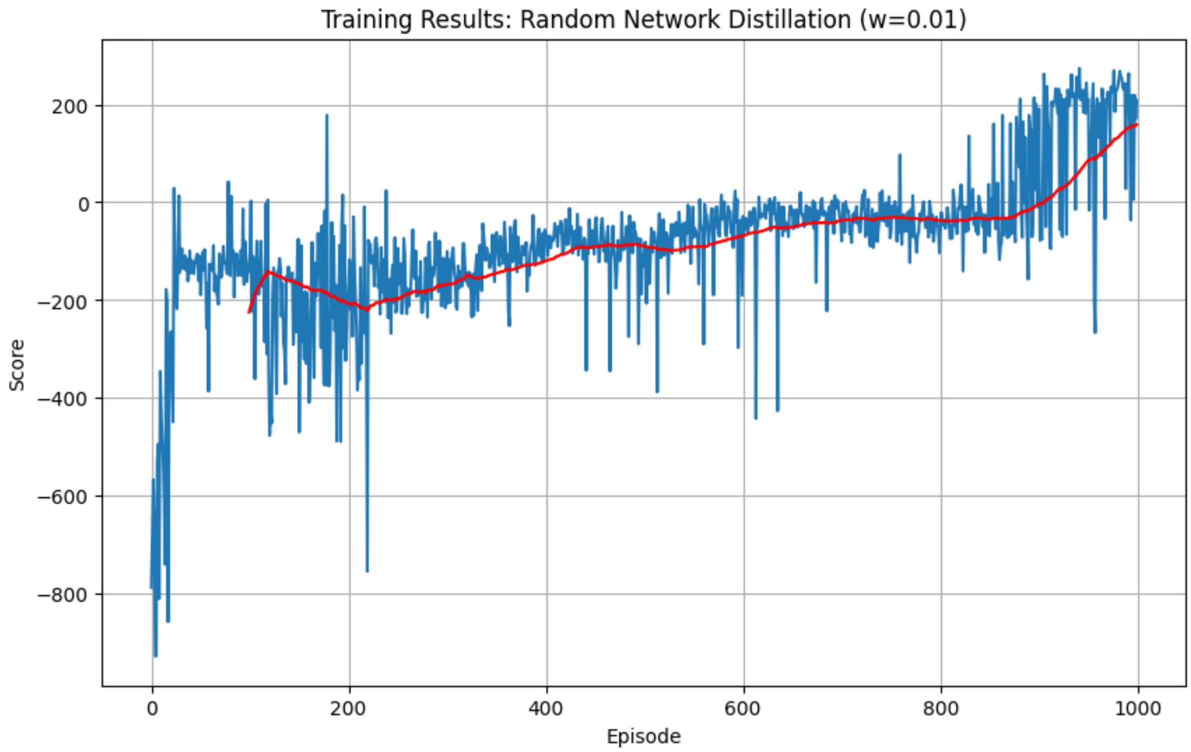
method and debug.

In [14]:
```
def test_individual_method(agent_class, agent_params=None, n_episodes=
    """
    Test a single exploration method to verify it works correctly.

    Parameters:
    -----------
    agent_class : class
        The agent class to instantiate
    agent_params : dict, optional
        Parameters to pass to the agent constructor
    n_episodes : int
        Number of episodes to train for
    max_t : int
        Maximum timesteps per episode
    print_every : int
        How often to print progress

    Returns:
    --------
    agent : instance
        The trained agent
    scores : list
        Training scores
    """
    # Set environment parameters
    env_name = 'LunarLander-v3'
    env = gym.make(env_name)
    state_size = env.observation_space.shape[0]
    action_size = env.action_space.n

    # Create agent with default or provided parameters
    if agent_params is None:
        agent_params = {}

    # Ensure seed is set
    if 'seed' not in agent_params:
        agent_params['seed'] = 42

    # Add learning_start if not specified
    if 'learning_start' not in agent_params:
        agent_params['learning_start'] = 1000

    # Create agent
    agent = agent_class(state_size, action_size, **agent_params)

    print(f"\n{'-'*50}")
    print(f"Testing agent: {agent.method_name}")
    print(f"{'-'*50}")

    # Train the agent
```

```
    scores = train_agent(agent, env_name=env_name, n_episodes=n_episod

    # Plot the scores
    plt.figure(figsize=(10, 6))
    plt.plot(np.arange(len(scores)), scores)

    # Add a rolling average
    window_size = min(100, len(scores)//5)
    rolling_mean = np.convolve(scores, np.ones(window_size)/window_siz
    plt.plot(np.arange(window_size-1, len(scores)), rolling_mean, 'r-'

    plt.title(f"Training Results: {agent.method_name}")
    plt.xlabel('Episode')
    plt.ylabel('Score')
    plt.grid(True)
    plt.show()

    return agent, scores

# Test Epsilon Greedy (Decaying)
def test_epsilon_greedy():
    params = {
        'eps_start': 1.0,
        'eps_end': 0.01,
        'eps_decay': 0.995,
        'learning_start': 1000
    }
    return test_individual_method(DQN_EpsilonGreedy, params)

# Test Fixed Epsilon
def test_fixed_epsilon():
    params = {
        'epsilon': 0.1,
        'learning_start': 1000
    }
    return test_individual_method(DQN_FixedEpsilon, params)

# Test Softmax
def test_softmax():
    params = {
        'initial_temperature':1.0,
        'min_temperature':0.01,
        'learning_start': 1000
    }
    return test_individual_method(DQN_Softmax, params)

# Test RND
def test_rnd():
    params = {
        'intrinsic_weight': 0.01,
        'learning_start': 1000
    }
```

```
    return test_individual_method(DQN_RND, params)

# Usage:
# agent, scores = test_epsilon_greedy()
# agent, scores = test_fixed_epsilon()
# agent, scores = test_softmax()
agent, scores = test_rnd()
```

```
--------------------------------------------------
Testing agent: Random Network Distillation (w=0.01)
--------------------------------------------------
  0%|          | 0/1000 [00:00<?, ?it/s]
Episode 100/1000 | Avg Score: -224.43 | Elapsed: 8.35s
Episode 200/1000 | Avg Score: -206.00 | Elapsed: 31.23s
Episode 300/1000 | Avg Score: -167.62 | Elapsed: 80.78s
Episode 400/1000 | Avg Score: -120.39 | Elapsed: 116.03s
Episode 500/1000 | Avg Score: -89.45 | Elapsed: 112.09s
Episode 600/1000 | Avg Score: -69.12 | Elapsed: 116.59s
Episode 700/1000 | Avg Score: -36.65 | Elapsed: 125.57s
Episode 800/1000 | Avg Score: -37.27 | Elapsed: 127.64s
Episode 900/1000 | Avg Score: -4.61 | Elapsed: 117.06s
Episode 1000/1000 | Avg Score: 158.99 | Elapsed: 78.66s
```



Now, if your implementations are sucessful, run the following code to benchmark these methods. We run each methods for 3 repeated trials.

In [15]:
```
def run_comparison(n_episodes=1200, n_trials=3, print_every=100):
    """Run a comparison of different DQN exploration methods."""
```

```
    # Set environment parameters
    env_name = 'LunarLander-v3'
    env = gym.make(env_name)
    state_size = env.observation_space.shape[0]
    action_size = env.action_space.n

    # Create agents with learning_start parameter
    learning_start = 1000  # Common value for all agents

    agents = [
        DQN_EpsilonGreedy(state_size, action_size, seed=0, learning_st
        DQN_FixedEpsilon(state_size, action_size, seed=0, epsilon=0.1,
        DQN_Softmax(state_size, action_size, seed=0, initial_temperatu
        DQN_RND(state_size, action_size, seed=0, intrinsic_weight=0.01
    ]

    # Run benchmark
    results_df, all_training_scores = benchmark_agents(
        agents,
        env_name=env_name,
        n_episodes=n_episodes,
        n_trials=n_trials,
        print_every=print_every
    )

    # Show results
    print("\nBenchmark Results:")
    print(results_df)

    # Plot results
    plot_benchmark_results(results_df)
    plot_training_curves(all_training_scores)

    return results_df, all_training_scores, agents


# Record videos of trained agents
def record_agent_videos(agents, env_name='LunarLander-v3'):
    """Record videos of trained agents."""
    os.makedirs('videos', exist_ok=True)

    for agent in agents:
        print(f"Recording {agent.method_name}...")
        env = gym.make(env_name, render_mode='rgb_array')
        env = RecordVideo(env, f"videos/{agent.method_name.replace(' '

        state, _ = env.reset(seed=0)
        done = False
        score = 0

        while not done:
            action = agent.act(state, training=False)
```

```
            next_state, reward, terminated, truncated, _ = env.step(ac
            done = terminated or truncated
            state = next_state
            score += reward

        print(f"Final score: {score}")
        env.close()
```

In [16]:
```
results, all_training_scores, trained_agents = run_comparison()
```

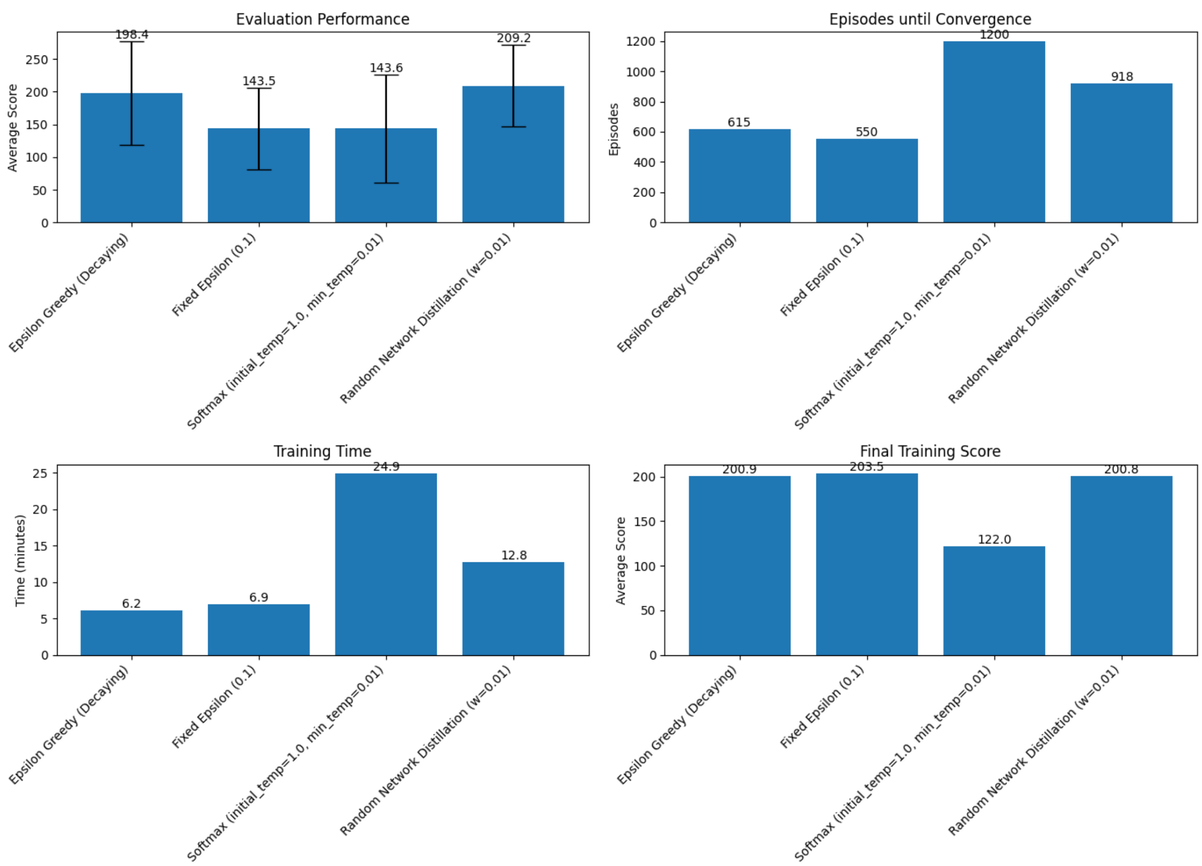
```
--------------------------------------------------
Training agent: Epsilon Greedy (Decaying)
--------------------------------------------------

Trial 1/3
  0%|          | 0/1200 [00:00<?, ?it/s]
Episode 100/1200 | Avg Score: -164.70 | Elapsed: 11.56s
Epsilon: 0.6058
Episode 200/1200 | Avg Score: -91.57 | Elapsed: 23.26s
Epsilon: 0.3670
Episode 300/1200 | Avg Score: -78.96 | Elapsed: 74.69s
Epsilon: 0.2223
Episode 400/1200 | Avg Score: 15.36 | Elapsed: 103.56s
Epsilon: 0.1347
Episode 500/1200 | Avg Score: 137.09 | Elapsed: 71.61s
Epsilon: 0.0816
Episode 600/1200 | Avg Score: 195.36 | Elapsed: 54.34s
Epsilon: 0.0494

Environment solved in 615 episodes!     Average Score: 201.19

Trial 2/3
  0%|          | 0/1200 [00:00<?, ?it/s]
Episode 100/1200 | Avg Score: -154.57 | Elapsed: 11.53s
Epsilon: 0.6058
Episode 200/1200 | Avg Score: -73.55 | Elapsed: 24.87s
Epsilon: 0.3670
Episode 300/1200 | Avg Score: -35.59 | Elapsed: 80.39s
Epsilon: 0.2223
Episode 400/1200 | Avg Score: -25.76 | Elapsed: 101.38s
Epsilon: 0.1347
Episode 500/1200 | Avg Score: -8.55 | Elapsed: 106.55s
Epsilon: 0.0816
Episode 600/1200 | Avg Score: 99.53 | Elapsed: 85.86s
Epsilon: 0.0494
Episode 700/1200 | Avg Score: 196.17 | Elapsed: 44.45s
Epsilon: 0.0299

Environment solved in 702 episodes!     Average Score: 201.31

Trial 3/3
  0%|          | 0/1200 [00:00<?, ?it/s]
```

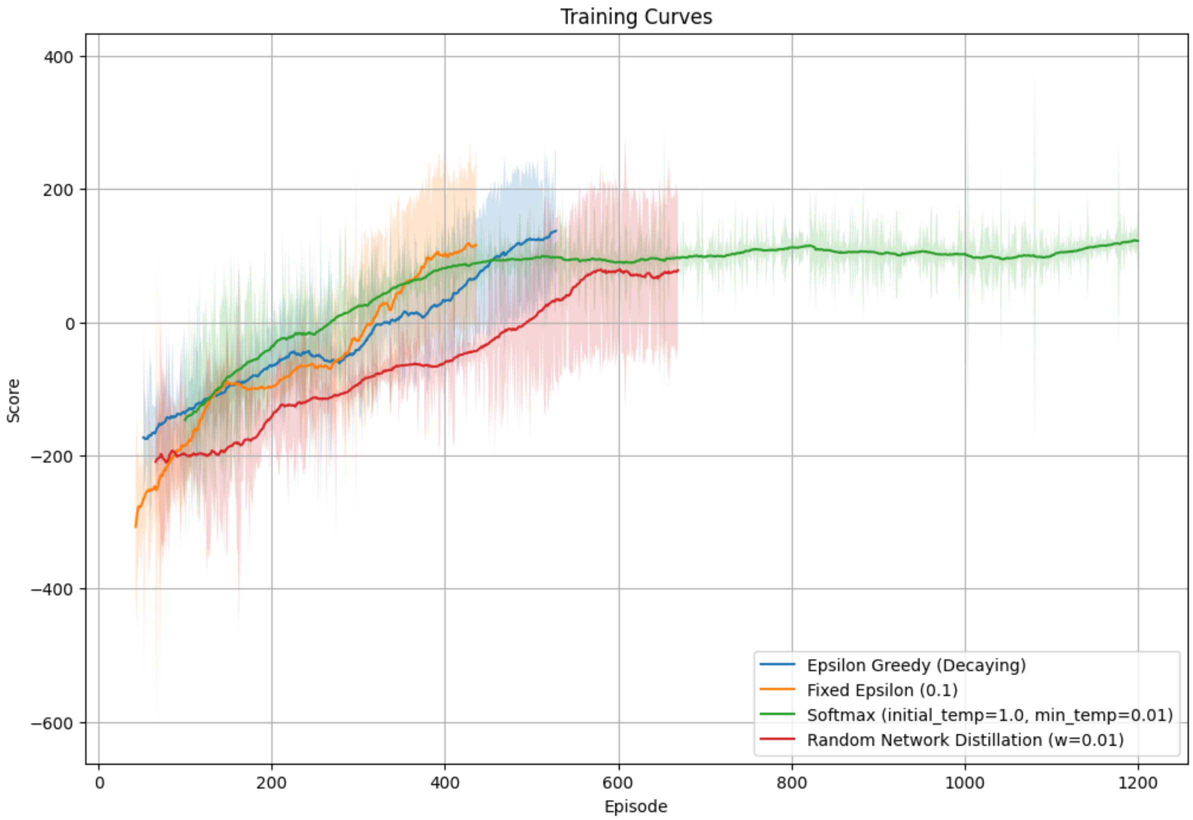
```
Episode 100/1200 | Avg Score: -147.65 | Elapsed: 13.01s
Epsilon: 0.6058
Episode 200/1200 | Avg Score: -73.19 | Elapsed: 28.15s
Epsilon: 0.3670
Episode 300/1200 | Avg Score: -24.16 | Elapsed: 90.50s
Epsilon: 0.2223
Episode 400/1200 | Avg Score: 75.94 | Elapsed: 91.67s
Epsilon: 0.1347
Episode 500/1200 | Avg Score: 187.01 | Elapsed: 67.64s
Epsilon: 0.0816

Environment solved in 528 episodes!     Average Score: 200.11


--------------------------------------------------
Training agent: Fixed Epsilon (0.1)
--------------------------------------------------

Trial 1/3
  0%|          | 0/1200 [00:00<?, ?it/s]
Episode 100/1200 | Avg Score: -269.49 | Elapsed: 39.48s
Epsilon: 0.1000
Episode 200/1200 | Avg Score: -88.56 | Elapsed: 103.98s
Epsilon: 0.1000
Episode 300/1200 | Avg Score: -49.46 | Elapsed: 94.16s
Epsilon: 0.1000
Episode 400/1200 | Avg Score: 150.72 | Elapsed: 68.46s
Epsilon: 0.1000

Environment solved in 436 episodes!     Average Score: 203.42

Trial 2/3
  0%|          | 0/1200 [00:00<?, ?it/s]
Episode 100/1200 | Avg Score: -202.80 | Elapsed: 35.32s
Epsilon: 0.1000
Episode 200/1200 | Avg Score: -104.45 | Elapsed: 92.69s
Epsilon: 0.1000
Episode 300/1200 | Avg Score: -13.24 | Elapsed: 89.61s
Epsilon: 0.1000
Episode 400/1200 | Avg Score: 108.27 | Elapsed: 66.99s
Epsilon: 0.1000
Episode 500/1200 | Avg Score: 186.46 | Elapsed: 60.59s
Epsilon: 0.1000

Environment solved in 529 episodes!     Average Score: 203.83

Trial 3/3
  0%|          | 0/1200 [00:00<?, ?it/s]
```

```
Episode 100/1200 | Avg Score: -253.92 | Elapsed: 18.27s
Epsilon: 0.1000
Episode 200/1200 | Avg Score: -96.14 | Elapsed: 99.87s
Epsilon: 0.1000
Episode 300/1200 | Avg Score: -83.40 | Elapsed: 101.78s
Epsilon: 0.1000
Episode 400/1200 | Avg Score: -39.43 | Elapsed: 109.02s
Epsilon: 0.1000
Episode 500/1200 | Avg Score: -16.67 | Elapsed: 105.62s
Epsilon: 0.1000
Episode 600/1200 | Avg Score: 140.09 | Elapsed: 81.27s
Epsilon: 0.1000

Environment solved in 686 episodes!     Average Score: 203.14


--------------------------------------------------
Training agent: Softmax (initial_temp=1.0, min_temp=0.01)
--------------------------------------------------

Trial 1/3
  0%|          | 0/1200 [00:00<?, ?it/s]
Episode 100/1200 | Avg Score: -128.78 | Elapsed: 36.59s
Episode 200/1200 | Avg Score: -3.44 | Elapsed: 73.51s
Episode 300/1200 | Avg Score: 0.98 | Elapsed: 136.08s
Episode 400/1200 | Avg Score: 65.97 | Elapsed: 138.86s
Episode 500/1200 | Avg Score: 105.69 | Elapsed: 144.04s
Episode 600/1200 | Avg Score: 104.20 | Elapsed: 142.52s
Episode 700/1200 | Avg Score: 104.45 | Elapsed: 141.61s
Episode 800/1200 | Avg Score: 109.69 | Elapsed: 141.59s
Episode 900/1200 | Avg Score: 105.53 | Elapsed: 141.46s
Episode 1000/1200 | Avg Score: 104.97 | Elapsed: 146.06s
Episode 1100/1200 | Avg Score: 113.34 | Elapsed: 144.83s
Episode 1200/1200 | Avg Score: 124.51 | Elapsed: 144.89s

Trial 2/3
  0%|          | 0/1200 [00:00<?, ?it/s]
Episode 100/1200 | Avg Score: -105.03 | Elapsed: 33.99s
Episode 200/1200 | Avg Score: -32.83 | Elapsed: 113.95s
Episode 300/1200 | Avg Score: 21.20 | Elapsed: 131.29s
Episode 400/1200 | Avg Score: 85.65 | Elapsed: 132.38s
Episode 500/1200 | Avg Score: 77.44 | Elapsed: 137.73s
Episode 600/1200 | Avg Score: 84.97 | Elapsed: 136.55s
Episode 700/1200 | Avg Score: 92.03 | Elapsed: 137.21s
Episode 800/1200 | Avg Score: 111.06 | Elapsed: 135.28s
Episode 900/1200 | Avg Score: 103.60 | Elapsed: 128.66s
Episode 1000/1200 | Avg Score: 100.41 | Elapsed: 138.29s
Episode 1100/1200 | Avg Score: 81.68 | Elapsed: 133.34s
Episode 1200/1200 | Avg Score: 118.63 | Elapsed: 141.14s

Trial 3/3
  0%|          | 0/1200 [00:00<?, ?it/s]
```

```
Episode 100/1200 | Avg Score: -206.35 | Elapsed: 28.08s
Episode 200/1200 | Avg Score: -72.97 | Elapsed: 63.25s
Episode 300/1200 | Avg Score: 50.91 | Elapsed: 133.96s
Episode 400/1200 | Avg Score: 91.06 | Elapsed: 137.47s
Episode 500/1200 | Avg Score: 102.39 | Elapsed: 139.02s
Episode 600/1200 | Avg Score: 80.77 | Elapsed: 129.60s
Episode 700/1200 | Avg Score: 97.79 | Elapsed: 135.82s
Episode 800/1200 | Avg Score: 115.64 | Elapsed: 138.58s
Episode 900/1200 | Avg Score: 100.55 | Elapsed: 127.15s
Episode 1000/1200 | Avg Score: 101.74 | Elapsed: 136.44s
Episode 1100/1200 | Avg Score: 101.95 | Elapsed: 136.89s
Episode 1200/1200 | Avg Score: 122.98 | Elapsed: 139.67s


--------------------------------------------------
Training agent: Random Network Distillation (w=0.01)
--------------------------------------------------

Trial 1/3
  0%|          | 0/1200 [00:00<?, ?it/s]
Episode 100/1200 | Avg Score: -220.79 | Elapsed: 25.03s
Episode 200/1200 | Avg Score: -143.23 | Elapsed: 70.62s
Episode 300/1200 | Avg Score: -57.50 | Elapsed: 120.95s
Episode 400/1200 | Avg Score: -36.70 | Elapsed: 118.05s
Episode 500/1200 | Avg Score: 16.59 | Elapsed: 114.14s
Episode 600/1200 | Avg Score: 69.49 | Elapsed: 78.68s
Episode 700/1200 | Avg Score: 55.78 | Elapsed: 91.25s
Episode 800/1200 | Avg Score: 145.53 | Elapsed: 79.12s
Episode 900/1200 | Avg Score: 180.39 | Elapsed: 71.44s

Environment solved in 931 episodes!     Average Score: 201.68

Trial 2/3
  0%|          | 0/1200 [00:00<?, ?it/s]
Episode 100/1200 | Avg Score: -226.15 | Elapsed: 17.38s
Episode 200/1200 | Avg Score: -135.45 | Elapsed: 79.26s
Episode 300/1200 | Avg Score: -100.85 | Elapsed: 115.32s
Episode 400/1200 | Avg Score: -61.49 | Elapsed: 113.64s
Episode 500/1200 | Avg Score: 17.92 | Elapsed: 115.89s
Episode 600/1200 | Avg Score: 170.31 | Elapsed: 77.81s

Environment solved in 669 episodes!     Average Score: 200.35

Trial 3/3
  0%|          | 0/1200 [00:00<?, ?it/s]
```

```
Episode 100/1200 | Avg Score: -165.82 | Elapsed: 8.86s
Episode 200/1200 | Avg Score: -200.97 | Elapsed: 12.46s
Episode 300/1200 | Avg Score: -154.64 | Elapsed: 82.71s
Episode 400/1200 | Avg Score: -83.92 | Elapsed: 119.40s
Episode 500/1200 | Avg Score: -58.04 | Elapsed: 112.87s
Episode 600/1200 | Avg Score: -26.81 | Elapsed: 123.55s
Episode 700/1200 | Avg Score: -15.89 | Elapsed: 127.12s
Episode 800/1200 | Avg Score: 73.20 | Elapsed: 106.18s
Episode 900/1200 | Avg Score: 118.46 | Elapsed: 83.01s
Episode 1000/1200 | Avg Score: 173.71 | Elapsed: 75.10s
Episode 1100/1200 | Avg Score: 165.89 | Elapsed: 66.21s

Environment solved in 1155 episodes!    Average Score: 200.27

Benchmark Results:
                                  agent_name  training_episodes  \
0                 Epsilon Greedy (Decaying)         615.000000
1                       Fixed Epsilon (0.1)         550.333333
2  Softmax (initial_temp=1.0, min_temp=0.01)       1200.000000
3       Random Network Distillation (w=0.01)         918.333333

   final_avg_score  eval_avg_score  eval_std_score  training_time
0       200.870514      198.367946       79.353041     369.432633
1       203.463002      143.473011       62.784564     416.869999
2       122.039417      143.623518       82.360515    1492.604905
3       200.768029      209.248653       62.653563     766.360985
```





In [17]:
```
print("\nTrained Agents Information:")
for agent in trained_agents:
    print(f"Agent Method: {agent.method_name}")
```

```
Trained Agents Information:
Agent Method: Epsilon Greedy (Decaying)
Agent Method: Fixed Epsilon (0.1)
Agent Method: Softmax (initial_temp=1.0, min_temp=0.01)
Agent Method: Random Network Distillation (w=0.01)
```

In [18]:
```
# Now record videos for all the trained agents.
# This function will create a folder named 'videos' and save videos in
record_agent_videos(trained_agents, env_name='LunarLander-v3')
```

```
Recording Epsilon Greedy (Decaying)...
/gpfs/gibbs/project/sds685/shared/conda_envs/rl_course/lib/python3.11/s
ite-packages/gymnasium/wrappers/rendering.py:283: UserWarning: WARN: Ov
erwriting existing videos at /vast/palmer/home.grace/ark89/SDS685_Pset
3/videos/Epsilon_Greedy_(Decaying) folder (try specifying a different `
video_folder` for the `RecordVideo` wrapper if this is not desired)
  logger.warn(
Final score: -16.711953652278325
Recording Fixed Epsilon (0.1)...
/gpfs/gibbs/project/sds685/shared/conda_envs/rl_course/lib/python3.11/s
ite-packages/gymnasium/wrappers/rendering.py:283: UserWarning: WARN: Ov
erwriting existing videos at /vast/palmer/home.grace/ark89/SDS685_Pset
3/videos/Fixed_Epsilon_(0.1) folder (try specifying a different `video_
folder` for the `RecordVideo` wrapper if this is not desired)
  logger.warn(
```

```
Final score: 10.473647327035547
Recording Softmax (initial_temp=1.0, min_temp=0.01)...
/gpfs/gibbs/project/sds685/shared/conda_envs/rl_course/lib/python3.11/site-packages/gymnasium/wrappers/rendering.py:283: UserWarning: WARN: Overwriting existing videos at /vast/palmer/home.grace/ark89/SDS685_Pset3/videos/Softmax_(initial_temp=1.0,_min_temp=0.01) folder (try specifying a different `video_folder` for the `RecordVideo` wrapper if this is not desired)
  logger.warn(
Final score: 267.7968919646281
Recording Random Network Distillation (w=0.01)...
/gpfs/gibbs/project/sds685/shared/conda_envs/rl_course/lib/python3.11/site-packages/gymnasium/wrappers/rendering.py:283: UserWarning: WARN: Overwriting existing videos at /vast/palmer/home.grace/ark89/SDS685_Pset3/videos/Random_Network_Distillation_(w=0.01) folder (try specifying a different `video_folder` for the `RecordVideo` wrapper if this is not desired)
  logger.warn(
Final score: 235.13341639584218
```

In [ ]:

In [1]:
```
import math
import random
import matplotlib.pyplot as plt
```

In [2]:
```
import math
import random
import matplotlib.pyplot as plt

# Set random seed for reproducibility.
random.seed(42)

# Global parameter for terminal state length.
K = 5  # We'll focus on k=5 for this demonstration.

# ------------------ True Reward Tree Construction ------------------

# Global dictionary to store rewards for terminal states.
global_terminal_rewards = {}

def generate_terminal_rewards(k, gap=0.2):
    """
    Generate rewards for all terminal states (binary strings of length
    For each terminal state, set y = 10 * random.random().
    Then, find the maximum y value and add gap to that terminal state'
    Returns a dictionary mapping each terminal state to its final rewa
    """
    num_terminals = 2 ** k
    rewards = {}
    for i in range(num_terminals):
        state = format(i, f"0{k}b")
        y = 2* random.random()
        rewards[state] = y
    max_y = max(rewards.values())
    # Add gap to the terminal state(s) that achieve the maximum reward
    for state in rewards:
        if abs(rewards[state] - max_y) < 1e-9:
            rewards[state] += gap
    return rewards

def reward_function(state):
    """Return the reward for a terminal state (binary string of length
    return global_terminal_rewards[state]

def build_full_tree(k):
    """
    Build the full binary tree for k (each terminal node is a binary s
    Returns a dictionary mapping state -> {'children': [child_state_1,
    For terminal nodes, the reward is taken from reward_function.
    For internal nodes, the reward is defined as the maximum of its ch
    """
```

```
    tree = {}
    def build_node(state):
        if len(state) == k:
            r = reward_function(state)
            tree[state] = {'children': [], 'reward': r}
            return r
        left = build_node(state + '0')
        right = build_node(state + '1')
        tree[state] = {'children': [state + '0', state + '1'], 'reward
        return tree[state]['reward']
    build_node("")  # Start from the root (empty string).
    return tree

def find_true_optimal_path(tree):
    """
    Traverse the full tree (dict) from the root (empty string) to the
    At each internal node, choose the child with the maximum reward.
    """
    optimal_path = []
    state = ""
    while True:
        optimal_path.append(state if state != "" else "root")
        if state not in tree or tree[state]['children'] == []:
            break
        children = tree[state]['children']
        if tree[children[0]]['reward'] >= tree[children[1]]['reward']:
            state = children[0]
        else:
            state = children[1]
    return optimal_path, tree[state]['reward']

def plot_true_tree(tree, k, optimal_path):
    """
    Plot the full binary tree for k using a manual layout.
    The x-coordinate is computed as (int(state,2)+0.5)/(2^(len(state))
    Terminal nodes are at level k. The optimal_path (a list) is used t
    """
    pos = {}
    labels = {}
    for state in tree:
        level = len(state)
        if state == "":
            x = 0.5
            label = "root"
        else:
            x = (int(state, 2) + 0.5) / (2 ** level)
            label = state
        y = level
        pos[label] = (x, -y)
        r = tree[state]['reward']
        labels[label] = f"{label}\nR:{r:.2f}"
```

```
    edges = []
    for state, node in tree.items():
        label = state if state != "" else "root"
        for child in node['children']:
            child_label = child if child != "" else "root"
            edges.append((label, child_label))

    plt.figure(figsize=(10, 8))
    for (u, v) in edges:
        plt.plot([pos[u][0], pos[v][0]], [pos[u][1], pos[v][1]], 'k-',
    for n, (x, y) in pos.items():
        color = 'yellow' if n in optimal_path else 'lightgreen'
        edge_color = 'red' if n in optimal_path else 'black'
        plt.scatter(x, y, s=1000, c=color, edgecolors=edge_color, zord
        plt.text(x, y, labels[n], horizontalalignment='center', vertic
    plt.title(f"True Reward Tree (k={k}) with Optimal Path Highlighted
    plt.axis('off')
    plt.show()
```

Implement MCTS

You are going to implement the methods that does best-child selection and backpropagation.

Best Child Function

This function implements the **child selection** mechanism of MCTS using the Upper Confidence Bound (UCB) formula. For each child node, a score is computed based on two components:

1. **Exploitation Term:**
   This is the average reward (also called the Q-value) for the child. It is computed as:

$$\text{avg\\_reward} = \frac{\text{child.reward}}{\text{child.visits}}$$

   This term reflects the observed performance (quality) of the child.

2. **Exploration Term:**
   This encourages choosing child nodes that have been less frequently explored. The exploration term is:

$$c \cdot \sqrt{\frac{2 \cdot \ln(\text{parent.visits})}{\text{child.visits}}}$$

where c is a parameter 'c_param' controlling the balance between exploration and exploitation.

The overall score for each child is the sum of its average reward and its exploration bonus. The function then selects the child node with the highest combined score.

If a child has never been visited ($\text{child.visits} = 0$), its score is set to infinity so that it will be selected for exploration.

Mathematical Explanation:

For a given parent node s and a child a among the set $A(s)$ of children, the UCB score is computed as follows:

1. **If** $N(a) = 0$ (the child has not been visited):

$$\text{Score}(a) = +\infty$$

2. **Otherwise**, let

$$Q(a) = \frac{R(a)}{N(a)}$$

   be the average reward of child a. Then, the UCB formula is given by:

$$\text{Score}(a) = Q(a) + c \cdot \sqrt{\frac{2 \cdot \ln(N(s))}{N(a)}}$$

where:

- $N(s)$ is the number of visits to the parent node,
- $N(a)$ is the number of visits to the child node,
- $R(a)$ is the cumulative reward for the child,
- c (here `c_param`) is a constant that regulates exploration versus exploitation.

Thus, the selection is:

$$a^* = \arg\max_{a \in A(s)} \left(\frac{R(a)}{N(a)} + c \cdot \sqrt{\frac{2 \cdot \ln(N(s))}{N(a)}} \right)$$

And the function returns the child a^* with the maximum score.

Backpropagation Function

This function implements the **backpropagation** step of the Monte Carlo Tree Search (MCTS) algorithm. After simulating a complete rollout (or playout) from a leaf node and obtaining a reward (which might be 1 if the simulation was successful, or 0 otherwise), the algorithm needs to update the nodes along the path **from that leaf up to the root** (how to traverse from leaf to the root?).

For each node on this path, the function:

- **Increments the visit count:** This tracks how many times that node (or state) has been encountered during the search.
- **Accumulates the reward:** It adds the simulation's reward to a running total stored in the node. This accumulated reward is later used to compute the average reward for each node.
- **Moves Up the Tree:** The process then continues to the node's parent until it reaches the root (i.e., until `node` is `None`).

Mathematical Explanation:

Suppose for each node s along the search path we store:

- $N(s)$: the number of visits to node s, (`node.visit`)
- $R(s)$: the cumulative reward obtained from simulations that passed through s (`node.reward`).

When a new simulation with reward r is completed, backpropagation updates every node s on the path as follows:

$$N(s) \leftarrow N(s) + 1, \qquad R(s) \leftarrow R(s) + r$$

Thus, along the entire path from the leaf to the root, the updates can be mathematically represented as:

$$\forall s \in \text{path}: \quad \begin{cases} N(s) = N(s) + 1, \\ R(s) = R(s) + r. \end{cases}$$

In [3]:
```
# ------------------ MCTS Implementation ------------------

class MCTS_Node:
    def __init__(self, state, parent=None):
        self.state = state        # Binary string representing the sta
        self.parent = parent      # Parent node.
        self.children = []        # List of child nodes.
        self.visits = 0           # Number of visits.
        self.reward = 0.0         # Cumulative reward from simulations
        self.untried_actions = self.get_possible_actions()
```

```
    def get_possible_actions(self):
        if len(self.state) >= K:
            return []
        return ['0', '1']

    def is_terminal(self):
        return len(self.state) >= K

    def is_fully_expanded(self):
        return len(self.untried_actions) == 0

    def best_child(self, c_param=5):
        best_score = -float('inf')
        best_child = None
        for child in self.children:
            if child.visits == 0:
                score = float('inf')
            else:
                avg_reward = child.reward / child.visits
                score = avg_reward + c_param * math.sqrt(math.log(self
            if score > best_score:
                best_score = score
                best_child = child
        return best_child

def mcts_tree_policy(node):
    """Selection and expansion: traverse until a node that is not full
    while not node.is_terminal():
        if not node.is_fully_expanded():
            action = node.untried_actions.pop()
            new_state = node.state + action
            child = MCTS_Node(new_state, parent=node)
            node.children.append(child)
            return child
        else:
            node = node.best_child()
    return node

def mcts_default_policy(state):
    """Random playout until a terminal state is reached."""
    current_state = state
    while len(current_state) < K:
        current_state += random.choice(['0', '1'])
    return reward_function(current_state)

def mcts_backup(node, reward):
    """Backpropagation: update the visit count and cumulative reward a
    while node is not None:
        node.visits += 1
        node.reward += reward
        node = node.parent
```

In [4]:
```
# ------------------ Manual Plotting of the MCTS Tree ----------------

def compute_node_position(state, k_value):
    """
    Compute (x,y) coordinates for a node based on its binary state and
    For a node at level L (L = len(state)):
         x = (int(state,2)+0.5)/(2^L) if state != "", else 0.5 for the
         y = -L.
    """
    if state == "":
        level = 0
        x = 0.5
    else:
        level = len(state)
        x = (int(state, 2) + 0.5) / (2 ** level)
    y = -level
    return x, y


def find_mcts_optimal_path(root):
    """
    Traverse from the root using best_child (with c_param=0) until a t
    Return the optimal path (list of state labels) and the average rew
    """
    path = []
    node = root
    while not node.is_terminal() and node.children:
        label = node.state if node.state != "" else "root"
        path.append(label)
        node = node.best_child(c_param=0)  # Greedy selection.
    label = node.state if node.state != "" else "root"
    path.append(label)
    avg_reward = node.reward / node.visits if node.visits > 0 else 0
    return path, avg_reward


def mcts_plot_tree(root, k_value):
    """
    Manually plot the MCTS tree using the node's state and level infor
    Highlight the optimal MCTS path.
    Labels for each node show the average reward ("avg").
    """
    pos = {}
    labels = {}
    nodes = {}

    def dfs(node):
        label = node.state if node.state != "" else "root"
        nodes[label] = node
        level = len(node.state)
        pos[label] = compute_node_position(node.state, k_value)
        avg = node.reward / node.visits if node.visits > 0 else 0
```

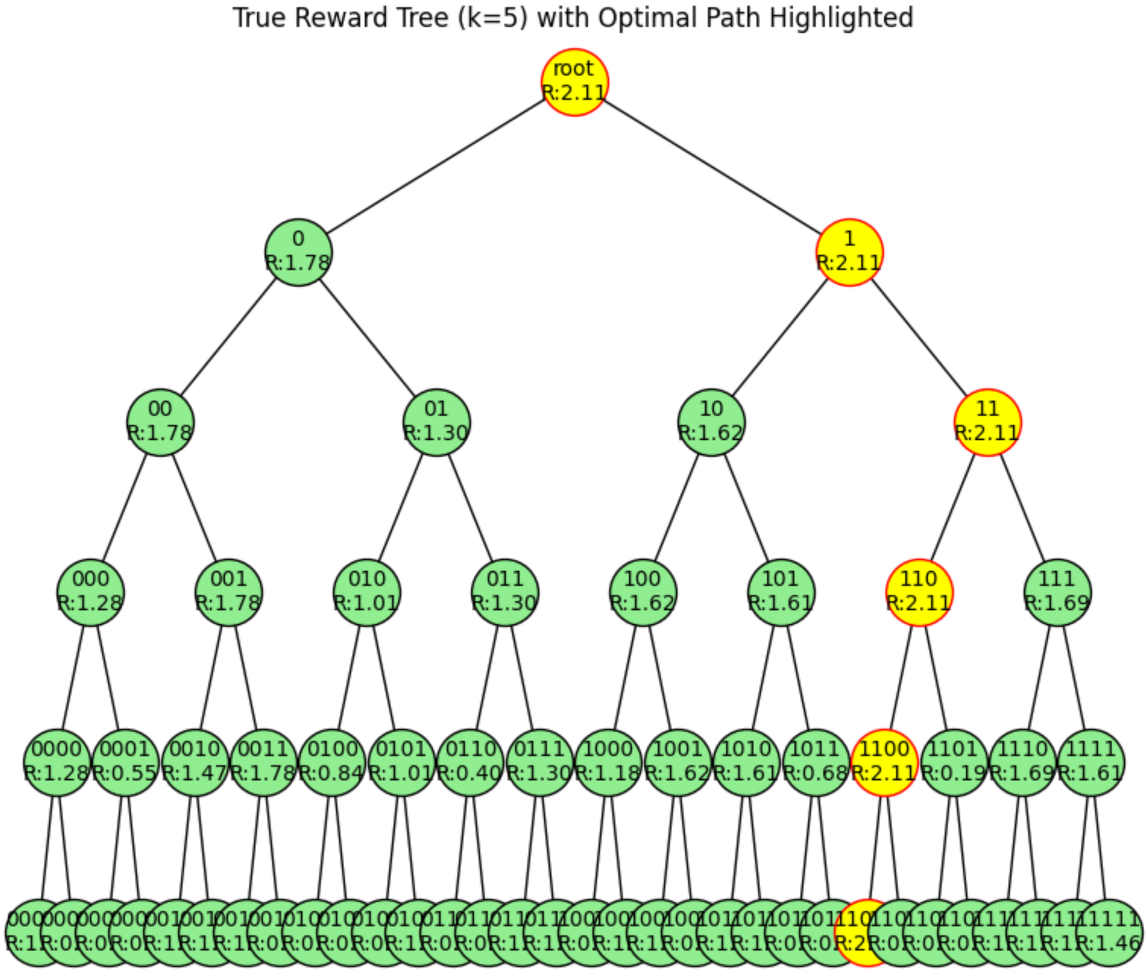
```
        labels[label] = f"{label}\navg:{avg:.2f}"
        for child in node.children:
            dfs(child)
    dfs(root)

    opt_path, _ = find_mcts_optimal_path(root)

    edges = []
    for label, node in nodes.items():
        if node.parent is not None:
            parent_label = node.parent.state if node.parent.state != "
            edges.append((parent_label, label))

    plt.figure(figsize=(10, 8))
    for (u, v) in edges:
        plt.plot([pos[u][0], pos[v][0]], [pos[u][1], pos[v][1]], 'k-',
    for n, (x, y) in pos.items():
        if n in opt_path:
            plt.scatter(x, y, s=1000, c='yellow', edgecolors='red', zo
        else:
            plt.scatter(x, y, s=1000, c='lightblue', edgecolors='black
        plt.text(x, y, labels[n], horizontalalignment='center', vertic
    plt.title(f"MCTS Tree (k={k_value}) with Optimal Path Highlighted"
    plt.axis('off')
    plt.show()
```

In [5]:
```
def mcts(root, iterations=100, plot_every=None, k_value=5, generate_pl
    """
    Run MCTS for a given number of iterations.
    If plot_every is specified, then:
      - If generate_plot is True, plot the current tree using mcts_plo
      - Otherwise, print the current optimal MCTS path and average rew
    k_value sets the terminal state length (K).

    In this version, Gaussian noise with variance 0.1 is added to the
    """
    global K
    K = k_value  # Set global terminal state length.

    noise_std = 1  # Standard deviation for noise

    for i in range(iterations):
        leaf = mcts_tree_policy(root)
        if not leaf.is_terminal():
            base_reward = mcts_default_policy(leaf.state)
        else:
            base_reward = reward_function(leaf.state)
        # Add Gaussian noise to the realized reward.
        sim_reward = base_reward + random.gauss(0, noise_std)

        mcts_backup(leaf, sim_reward)
```

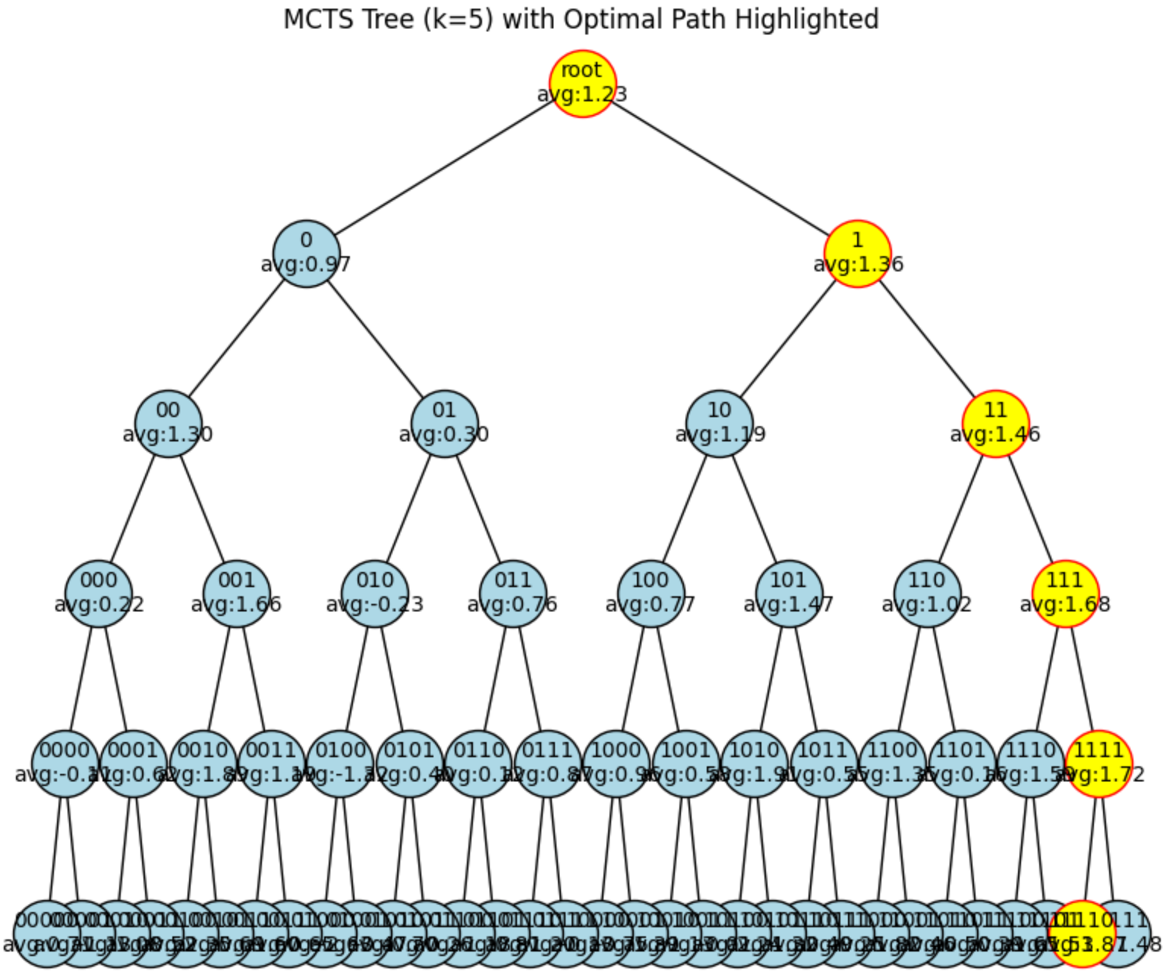
```
        if plot_every is not None and (i + 1) % plot_every == 0:
            opt_path, opt_reward = find_mcts_optimal_path(root)
            print(f"\nIteration {i+1}:")
            print("MCTS Optimal Path:", " -> ".join(opt_path))
            print(f"MCTS Optimal Reward: {opt_reward:.3f}")
            if generate_plot:
                mcts_plot_tree(root, k_value)
    return root
```

In [6]:
```
random.seed(42)
global K
# --- True Reward Tree for k = 5 ---
print("Building the True Reward Tree for k = 5...")
K = 5
global_terminal_rewards.clear()  # Clear previous rewards.
global_terminal_rewards.update(generate_terminal_rewards(K, gap=0.2))

# Print the rewards for all terminal states.
print(f"\nTerminal Rewards for k = {K}:")
num_terminals = 2 ** K
for i in range(num_terminals):
    state = format(i, f"0{K}b")
    print(f"State: {state} - Reward: {global_terminal_rewards[state]:.

# Build and plot the full true reward tree.
true_tree = build_full_tree(K)
true_opt_path, true_opt_reward = find_true_optimal_path(true_tree)
print("\nTrue Optimal Path:", " -> ".join(true_opt_path))
print(f"True Optimal Reward: {true_opt_reward:.3f}")
plot_true_tree(true_tree, K, true_opt_path)

# --- MCTS for k = 5 ---
print(f"\nRunning MCTS for k = {K}...")
mcts_root = MCTS_Node("")
iterations = 1000
plot_every = 200  # Plot or print optimal path every 100 iterations.
# Set generate_plot to False if you only want to print the optimal pat
mcts(mcts_root, iterations=iterations, plot_every=plot_every, k_value=
mcts_opt_path, mcts_opt_reward = find_mcts_optimal_path(mcts_root)
print(f"\n----- Finish MCTS for k = {K} -----")
print(f"\n----- Print the final results -----")
print("\nMCTS Optimal Path:", " -> ".join(mcts_opt_path))
print(f"MCTS Optimal Reward: {mcts_opt_reward:.3f}")
mcts_plot_tree(mcts_root, K)

# --- Structured Results ---

print("True Optimal Path:", " -> ".join(true_opt_path))
print("True Optimal Reward:", f"{true_opt_reward:.3f}")
print("MCTS Optimal Path:", " -> ".join(mcts_opt_path))
print("MCTS Optimal Reward:", f"{mcts_opt_reward:.3f}")
```

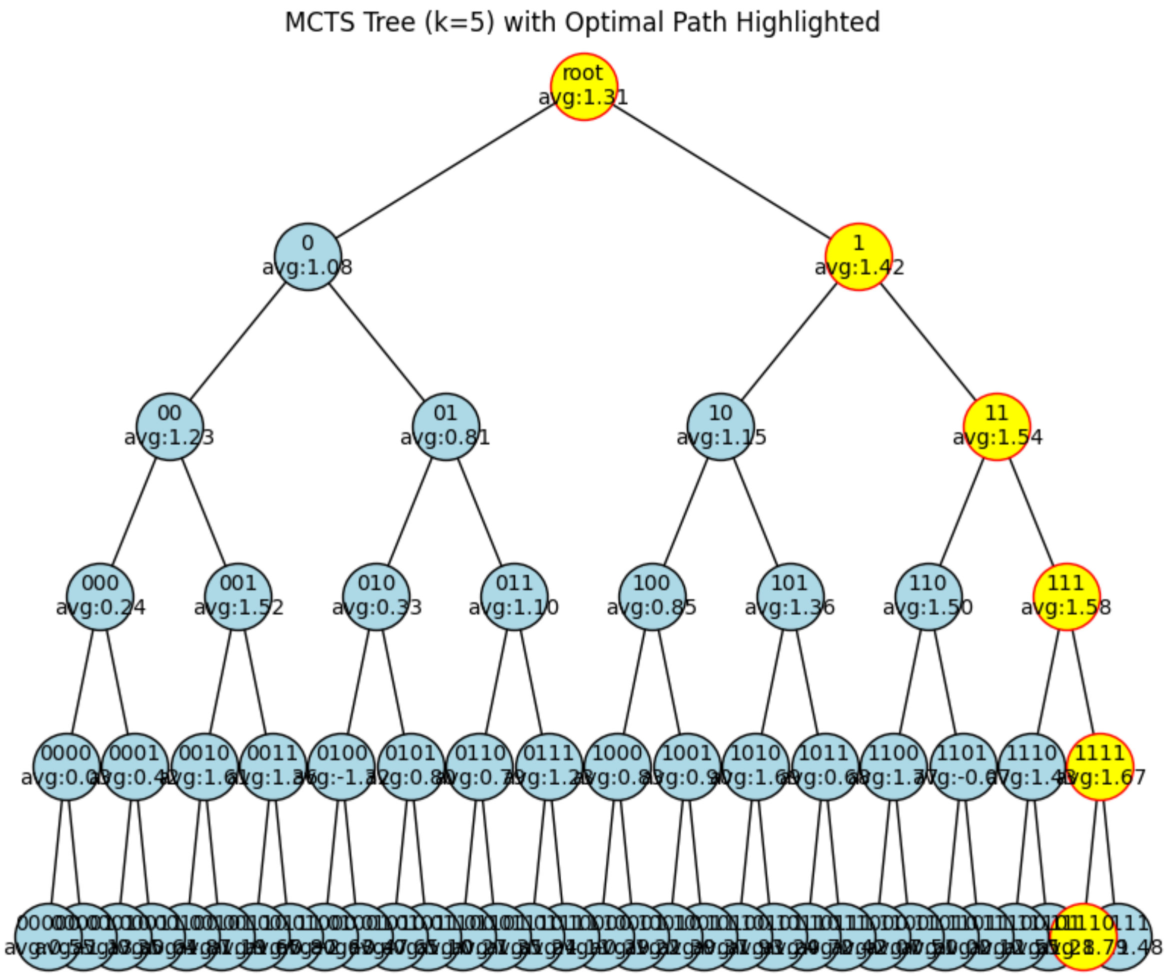
```
Building the True Reward Tree for k = 5...

Terminal Rewards for k = 5:
State: 00000 - Reward: 1.28
State: 00001 - Reward: 0.05
State: 00010 - Reward: 0.55
State: 00011 - Reward: 0.45
State: 00100 - Reward: 1.47
State: 00101 - Reward: 1.35
State: 00110 - Reward: 1.78
State: 00111 - Reward: 0.17
State: 01000 - Reward: 0.84
State: 01001 - Reward: 0.06
State: 01010 - Reward: 0.44
State: 01011 - Reward: 1.01
State: 01100 - Reward: 0.05
State: 01101 - Reward: 0.40
State: 01110 - Reward: 1.30
State: 01111 - Reward: 1.09
State: 10000 - Reward: 0.44
State: 10001 - Reward: 1.18
State: 10010 - Reward: 1.62
State: 10011 - Reward: 0.01
State: 10100 - Reward: 1.61
State: 10101 - Reward: 1.40
State: 10110 - Reward: 0.68
State: 10111 - Reward: 0.31
State: 11000 - Reward: 2.11
State: 11001 - Reward: 0.67
State: 11010 - Reward: 0.19
State: 11011 - Reward: 0.19
State: 11100 - Reward: 1.69
State: 11101 - Reward: 1.21
State: 11110 - Reward: 1.61
State: 11111 - Reward: 1.46

True Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000
True Optimal Reward: 2.114
```



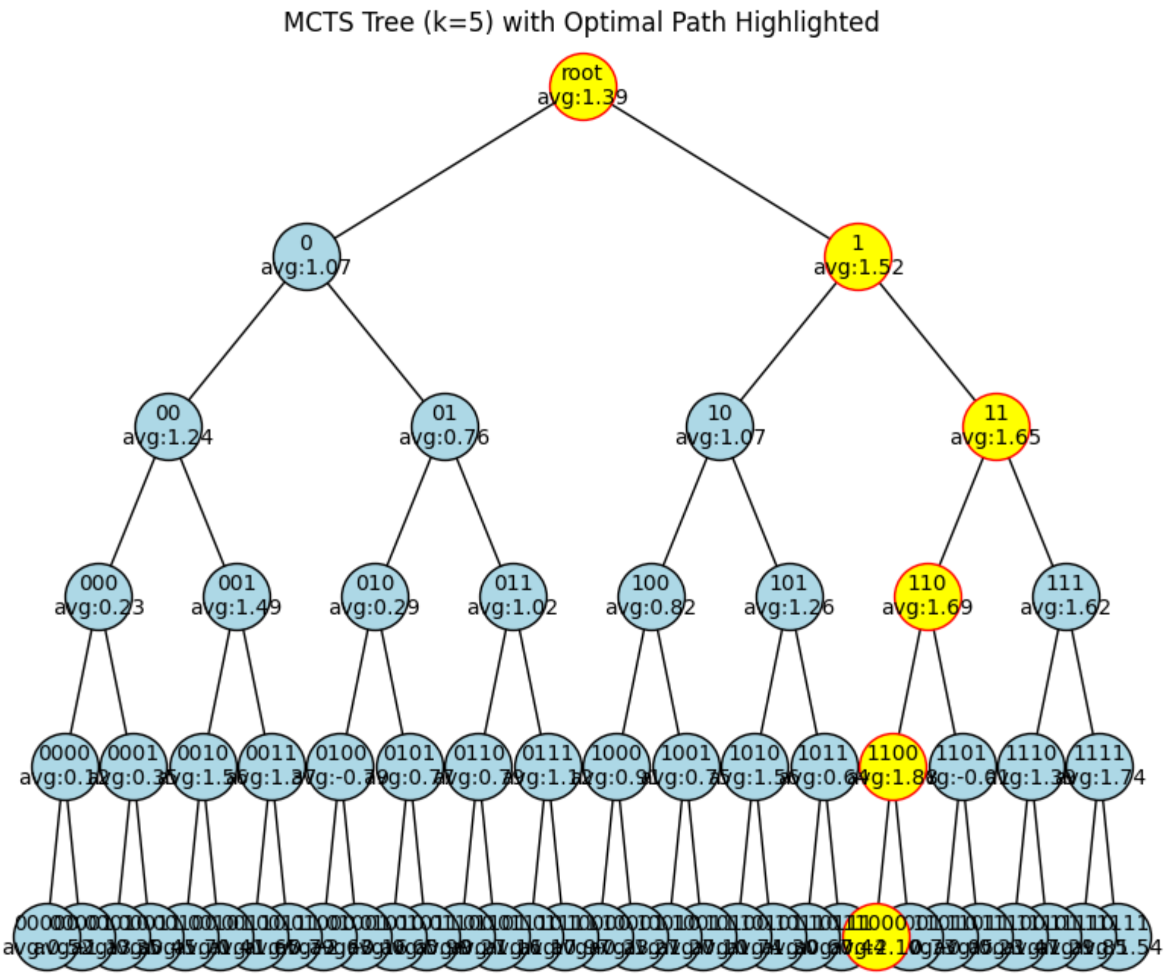

```
Running MCTS for k = 5...

Iteration 200:
MCTS Optimal Path: root -> 1 -> 11 -> 111 -> 1111 -> 11110
MCTS Optimal Reward: 1.871
```




```
Iteration 400:
MCTS Optimal Path: root -> 1 -> 11 -> 111 -> 1111 -> 11110
MCTS Optimal Reward: 1.792
```

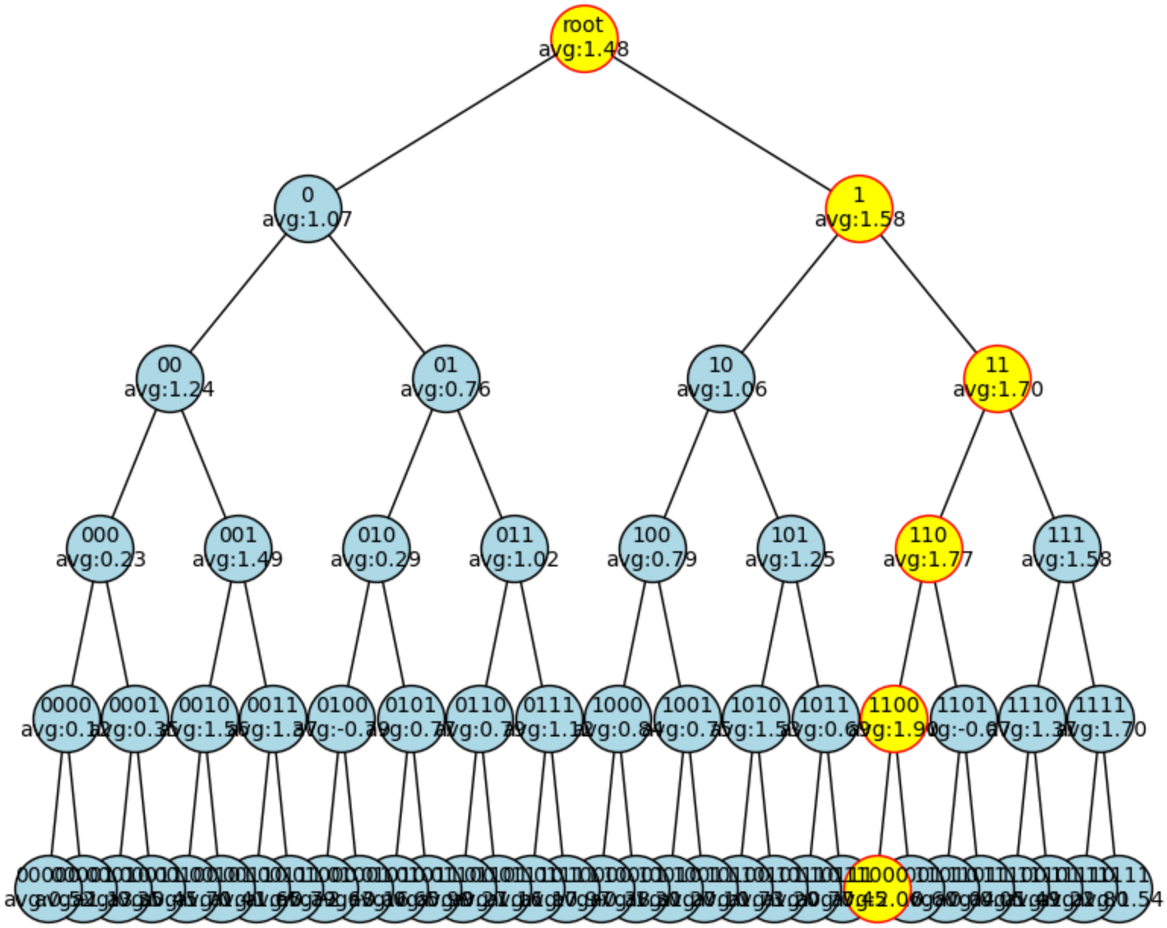



```
Iteration 600:
MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000
MCTS Optimal Reward: 2.095
```

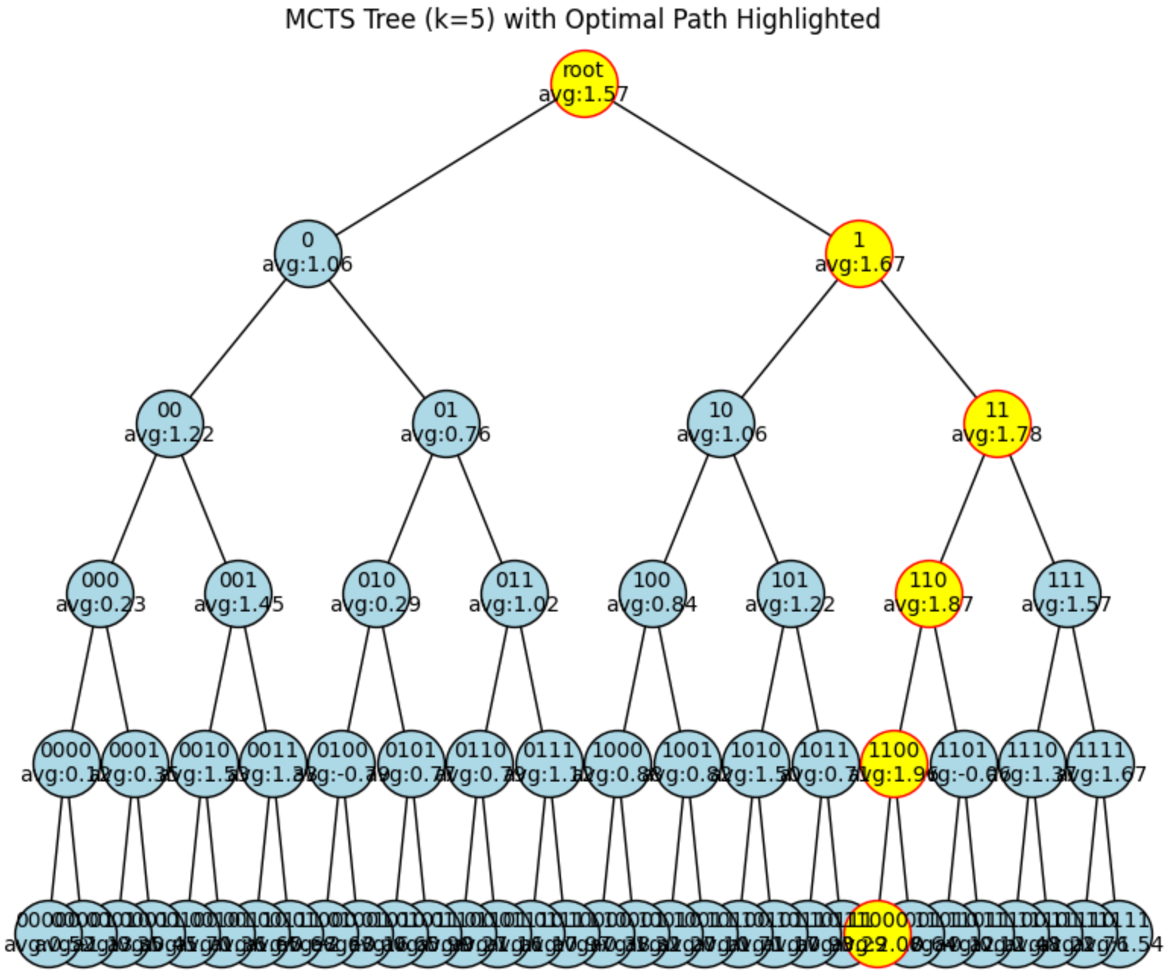



```
Iteration 800:
MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000
MCTS Optimal Reward: 2.060
```

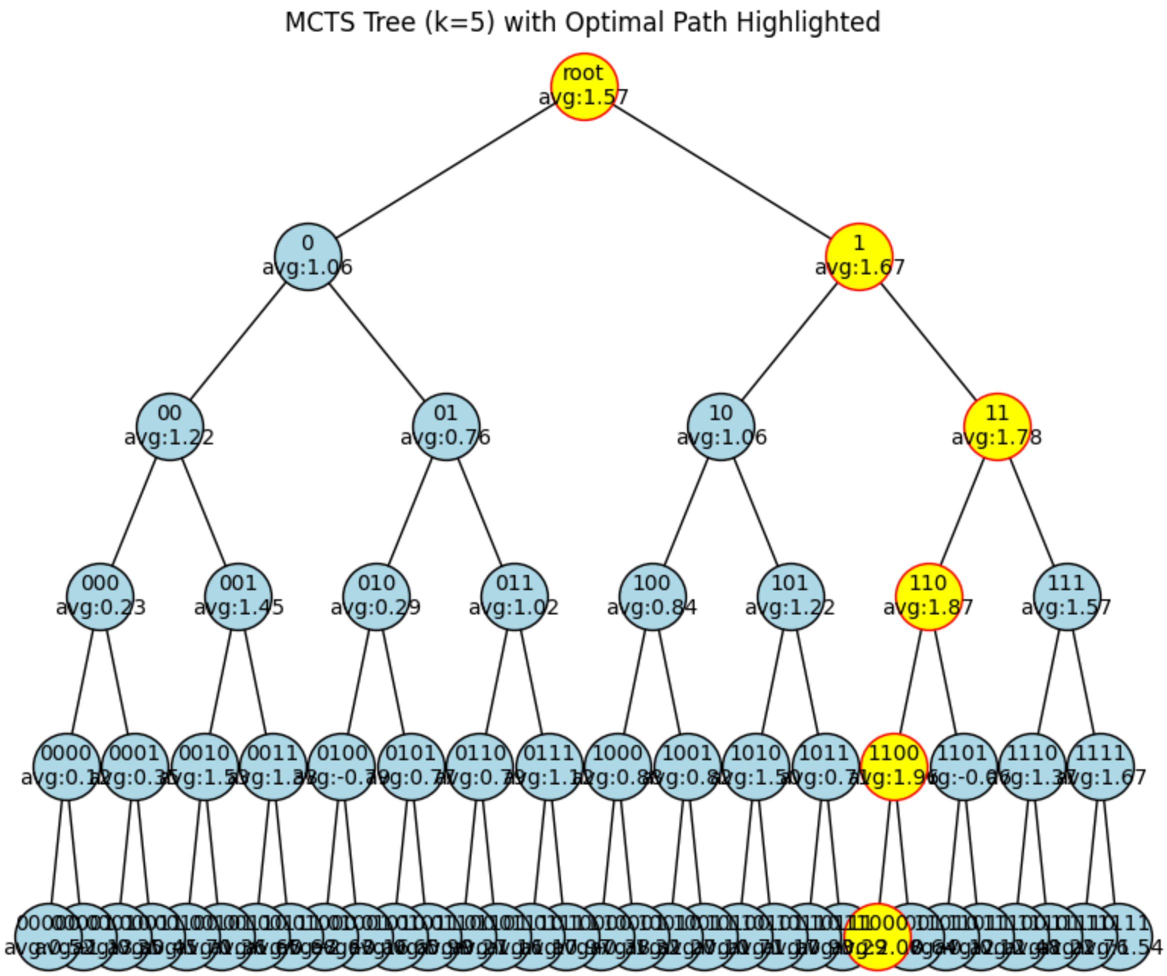
MCTS Tree (k=5) with Optimal Path Highlighted




```
Iteration 1000:
MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000
MCTS Optimal Reward: 2.082
```




```
----- Finish MCTS for k = 5 -----

----- Print the final results -----

MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000
MCTS Optimal Reward: 2.082
```




```
True Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000
True Optimal Reward: 2.114
MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000
MCTS Optimal Reward: 2.082
```

In [7]:

```python
random.seed(42)
global K
# --- True Reward Tree for k = 6 ---
K = 7
print(f"Building the True Reward Tree for k = {K}...")

global_terminal_rewards.clear()  # Clear previous rewards.
global_terminal_rewards.update(generate_terminal_rewards(K, gap=0.3))

# Print the rewards for all terminal states.
print(f"\nTerminal Rewards for k = {K}:")
num_terminals = 2 ** K
for i in range(num_terminals):
    state = format(i, f"0{K}b")
    print(f"State: {state} - Reward: {global_terminal_rewards[state]:.

# Build and plot the full true reward tree.
true_tree = build_full_tree(K)
true_opt_path, true_opt_reward = find_true_optimal_path(true_tree)
print("\nTrue Optimal Path:", " -> ".join(true_opt_path))
```

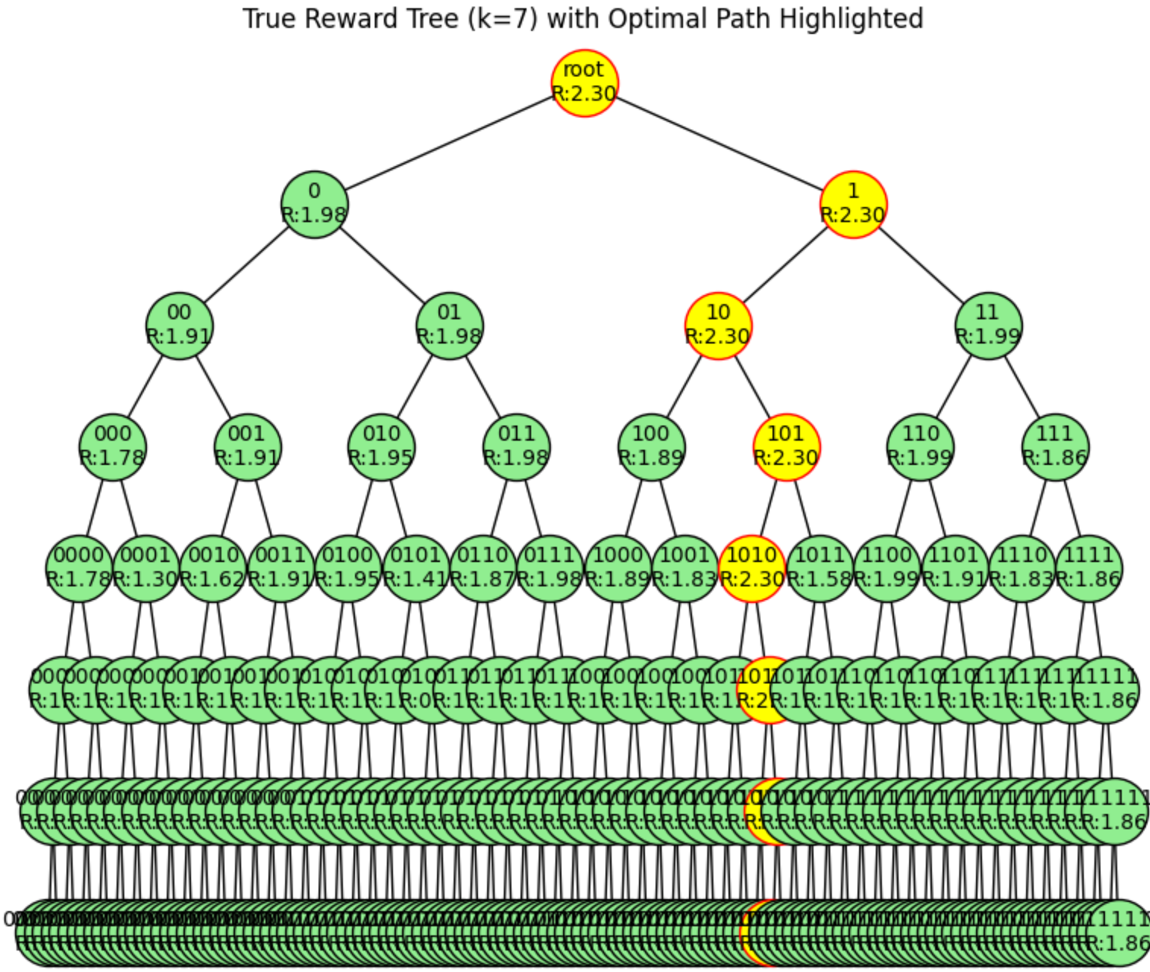
```python
print(f"True Optimal Reward: {true_opt_reward:.3f}")
plot_true_tree(true_tree, K, true_opt_path)

# --- MCTS for k = 7 ---
print(f"\nRunning MCTS for k = {K}...")
mcts_root = MCTS_Node("")
iterations = 2000
plot_every = 500
# Set generate_plot to False if you only want to print the optimal pat
mcts(mcts_root, iterations=iterations, plot_every=plot_every, k_value=
mcts_opt_path, mcts_opt_reward = find_mcts_optimal_path(mcts_root)
print(f"\n----- Finish MCTS for k = {K} -----")
print(f"\n----- Print the final results -----")
print("\nMCTS Optimal Path:", " -> ".join(mcts_opt_path))
print(f"MCTS Optimal Reward: {mcts_opt_reward:.3f}")
mcts_plot_tree(mcts_root, K)

# --- Structured Results ---

print("True Optimal Path:", " -> ".join(true_opt_path))
print("True Optimal Reward:", f"{true_opt_reward:.3f}")
print("MCTS Optimal Path:", " -> ".join(mcts_opt_path))
print("MCTS Optimal Reward:", f"{mcts_opt_reward:.3f}")
```

```
Building the True Reward Tree for k = 7...

Terminal Rewards for k = 7:
State: 0000000 - Reward: 1.28
State: 0000001 - Reward: 0.05
State: 0000010 - Reward: 0.55
State: 0000011 - Reward: 0.45
State: 0000100 - Reward: 1.47
State: 0000101 - Reward: 1.35
State: 0000110 - Reward: 1.78
State: 0000111 - Reward: 0.17
State: 0001000 - Reward: 0.84
State: 0001001 - Reward: 0.06
State: 0001010 - Reward: 0.44
State: 0001011 - Reward: 1.01
State: 0001100 - Reward: 0.05
State: 0001101 - Reward: 0.40
State: 0001110 - Reward: 1.30
State: 0001111 - Reward: 1.09
State: 0010000 - Reward: 0.44
State: 0010001 - Reward: 1.18
State: 0010010 - Reward: 1.62
State: 0010011 - Reward: 0.01
State: 0010100 - Reward: 1.61
State: 0010101 - Reward: 1.40
State: 0010110 - Reward: 0.68
State: 0010111 - Reward: 0.31
State: 0011000 - Reward: 1.91
State: 0011001 - Reward: 0.67
```

```
State: 0011010 - Reward: 0.19
State: 0011011 - Reward: 0.19
State: 0011100 - Reward: 1.69
State: 0011101 - Reward: 1.21
State: 0011110 - Reward: 1.61
State: 0011111 - Reward: 1.46
State: 0100000 - Reward: 1.07
State: 0100001 - Reward: 1.95
State: 0100010 - Reward: 0.76
State: 0100011 - Reward: 1.10
State: 0100100 - Reward: 1.66
State: 0100101 - Reward: 1.24
State: 0100110 - Reward: 1.72
State: 0100111 - Reward: 1.15
State: 0101000 - Reward: 1.41
State: 0101001 - Reward: 0.09
State: 0101010 - Reward: 0.46
State: 0101011 - Reward: 0.58
State: 0101100 - Reward: 0.16
State: 0101101 - Reward: 0.47
State: 0101110 - Reward: 0.20
State: 0101111 - Reward: 0.56
State: 0110000 - Reward: 1.27
State: 0110001 - Reward: 0.73
State: 0110010 - Reward: 0.74
State: 0110011 - Reward: 0.42
State: 0110100 - Reward: 0.53
State: 0110101 - Reward: 1.87
State: 0110110 - Reward: 1.30
State: 0110111 - Reward: 1.22
State: 0111000 - Reward: 0.34
State: 0111001 - Reward: 1.46
State: 0111010 - Reward: 0.33
State: 0111011 - Reward: 0.76
State: 0111100 - Reward: 1.98
State: 0111101 - Reward: 1.28
State: 0111110 - Reward: 1.11
State: 0111111 - Reward: 1.37
State: 1000000 - Reward: 1.69
State: 1000001 - Reward: 1.55
State: 1000010 - Reward: 0.46
State: 1000011 - Reward: 0.06
State: 1000100 - Reward: 0.63
State: 1000101 - Reward: 0.54
State: 1000110 - Reward: 0.42
State: 1000111 - Reward: 1.89
State: 1001000 - Reward: 1.75
State: 1001001 - Reward: 0.63
State: 1001010 - Reward: 1.31
State: 1001011 - Reward: 0.79
State: 1001100 - Reward: 1.83
State: 1001101 - Reward: 0.92
```

```
State: 1001110 - Reward: 0.53
State: 1001111 - Reward: 0.49
State: 1010000 - Reward: 1.12
State: 1010001 - Reward: 0.53
State: 1010010 - Reward: 1.17
State: 1010011 - Reward: 1.80
State: 1010100 - Reward: 0.80
State: 1010101 - Reward: 0.44
State: 1010110 - Reward: 2.30
State: 1010111 - Reward: 1.02
State: 1011000 - Reward: 0.18
State: 1011001 - Reward: 0.09
State: 1011010 - Reward: 0.22
State: 1011011 - Reward: 1.25
State: 1011100 - Reward: 1.58
State: 1011101 - Reward: 0.84
State: 1011110 - Reward: 0.13
State: 1011111 - Reward: 0.76
State: 1100000 - Reward: 1.99
State: 1100001 - Reward: 1.06
State: 1100010 - Reward: 1.94
State: 1100011 - Reward: 1.72
State: 1100100 - Reward: 0.02
State: 1100101 - Reward: 1.44
State: 1100110 - Reward: 1.36
State: 1100111 - Reward: 1.07
State: 1101000 - Reward: 0.53
State: 1101001 - Reward: 1.28
State: 1101010 - Reward: 0.22
State: 1101011 - Reward: 0.87
State: 1101100 - Reward: 0.91
State: 1101101 - Reward: 1.91
State: 1101110 - Reward: 1.75
State: 1101111 - Reward: 0.53
State: 1110000 - Reward: 1.00
State: 1110001 - Reward: 0.36
State: 1110010 - Reward: 1.83
State: 1110011 - Reward: 1.74
State: 1110100 - Reward: 0.60
State: 1110101 - Reward: 1.28
State: 1110110 - Reward: 1.22
State: 1110111 - Reward: 0.31
State: 1111000 - Reward: 1.53
State: 1111001 - Reward: 1.08
State: 1111010 - Reward: 1.56
State: 1111011 - Reward: 1.06
State: 1111100 - Reward: 0.00
State: 1111101 - Reward: 0.65
State: 1111110 - Reward: 0.04
State: 1111111 - Reward: 1.86

True Optimal Path: root -> 1 -> 10 -> 101 -> 1010 -> 10101 -> 101011 ->
```

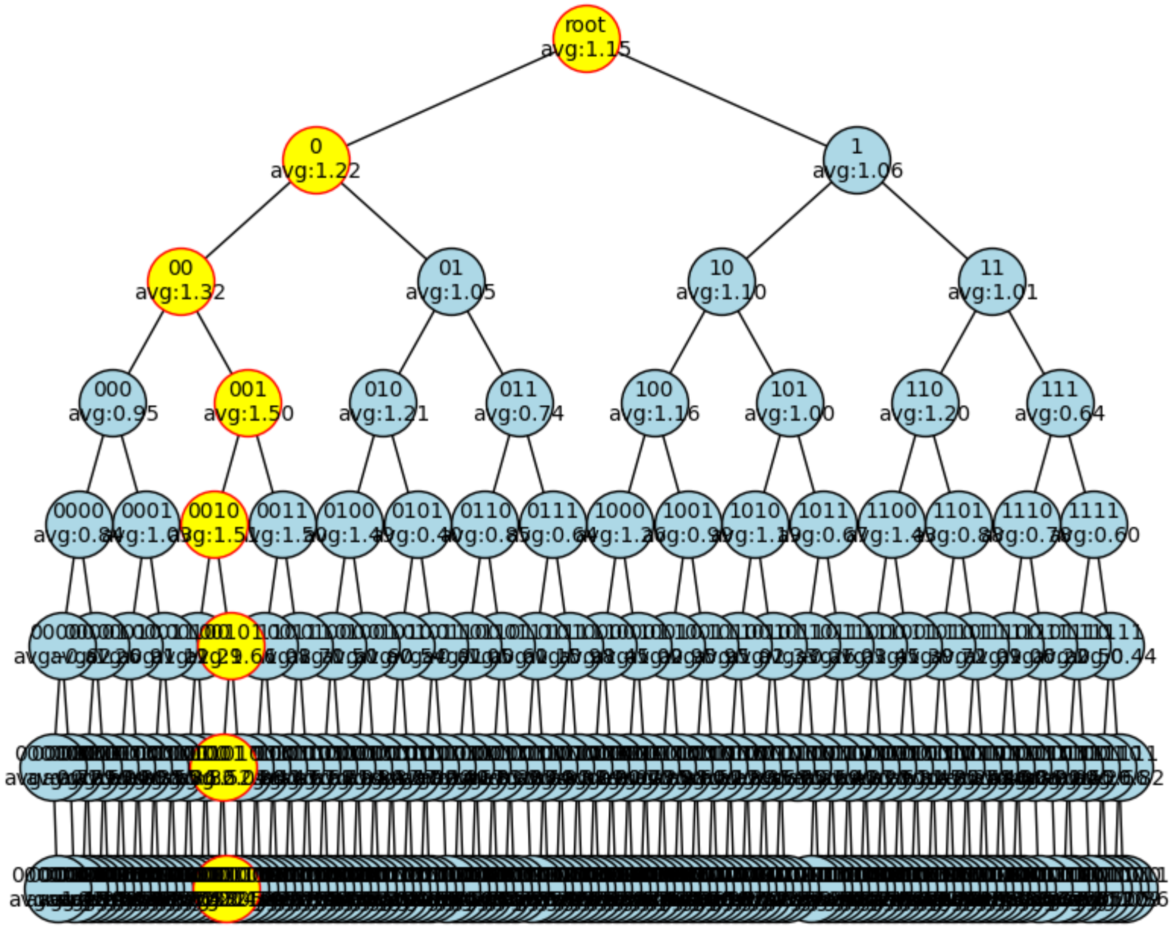
1010110
True Optimal Reward: 2.295



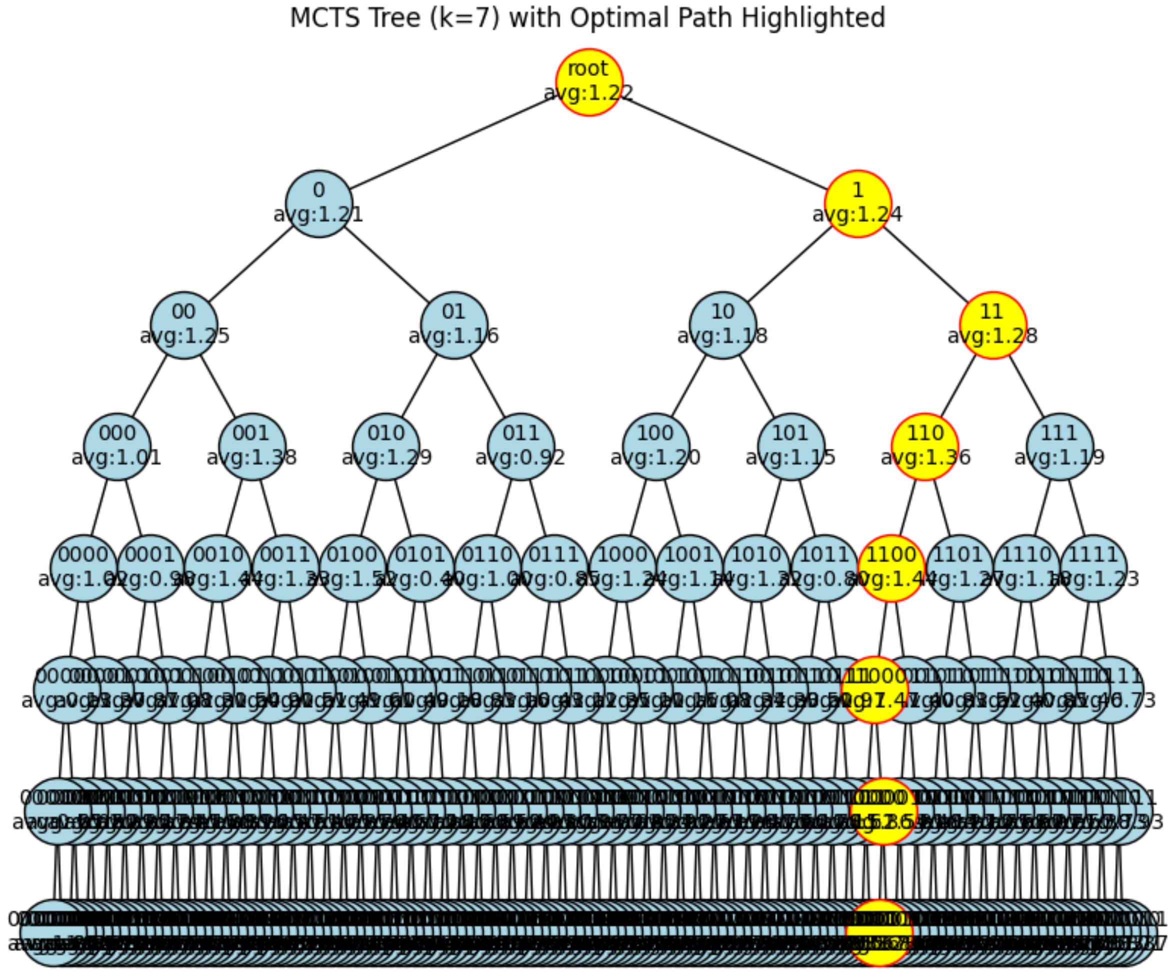
Running MCTS for k = 7...

Iteration 500:
MCTS Optimal Path: root -> 0 -> 00 -> 001 -> 0010 -> 00101 -> 001010 -> 0010101
MCTS Optimal Reward: 2.117

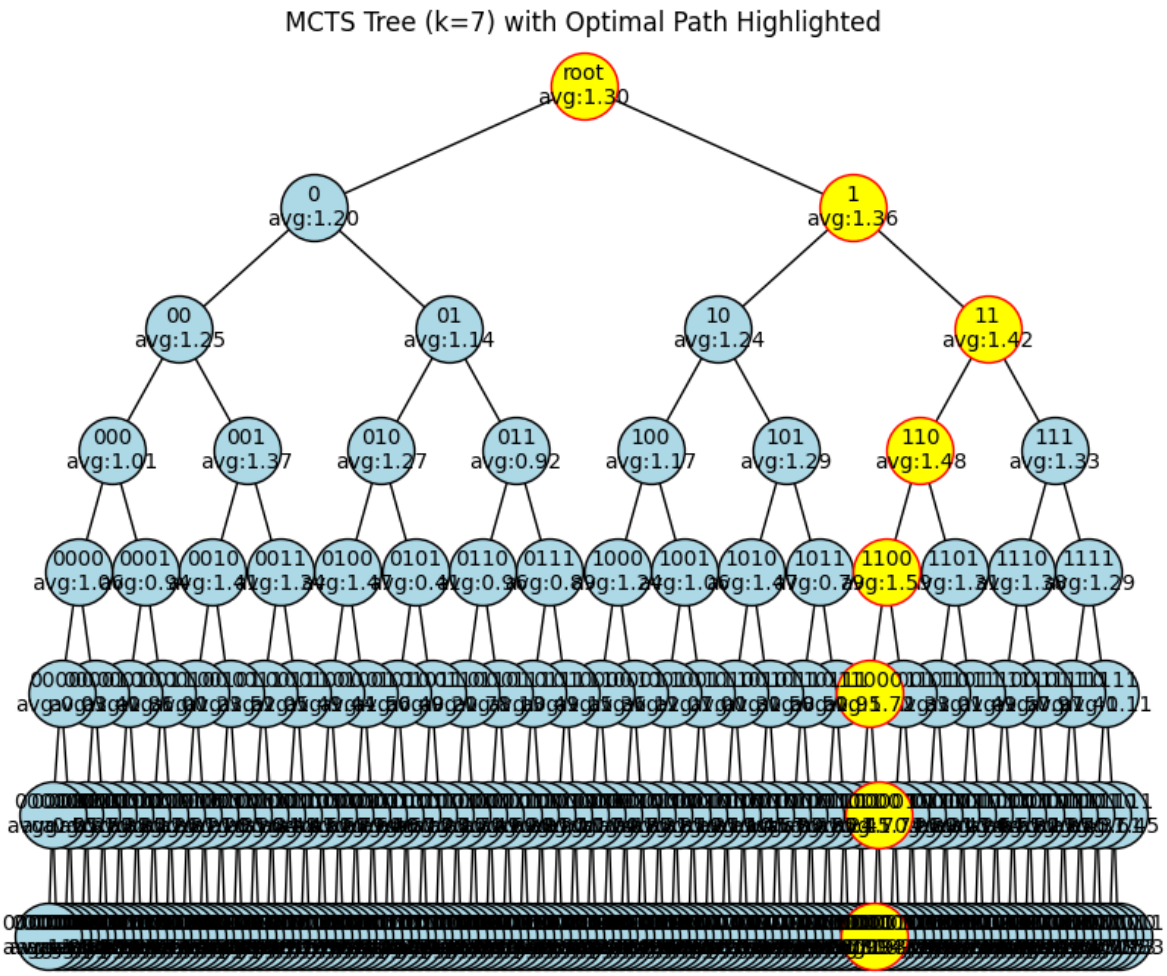
MCTS Tree (k=7) with Optimal Path Highlighted




```
Iteration 1000:
MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000 -> 110001 ->
1100010
MCTS Optimal Reward: 1.844
```

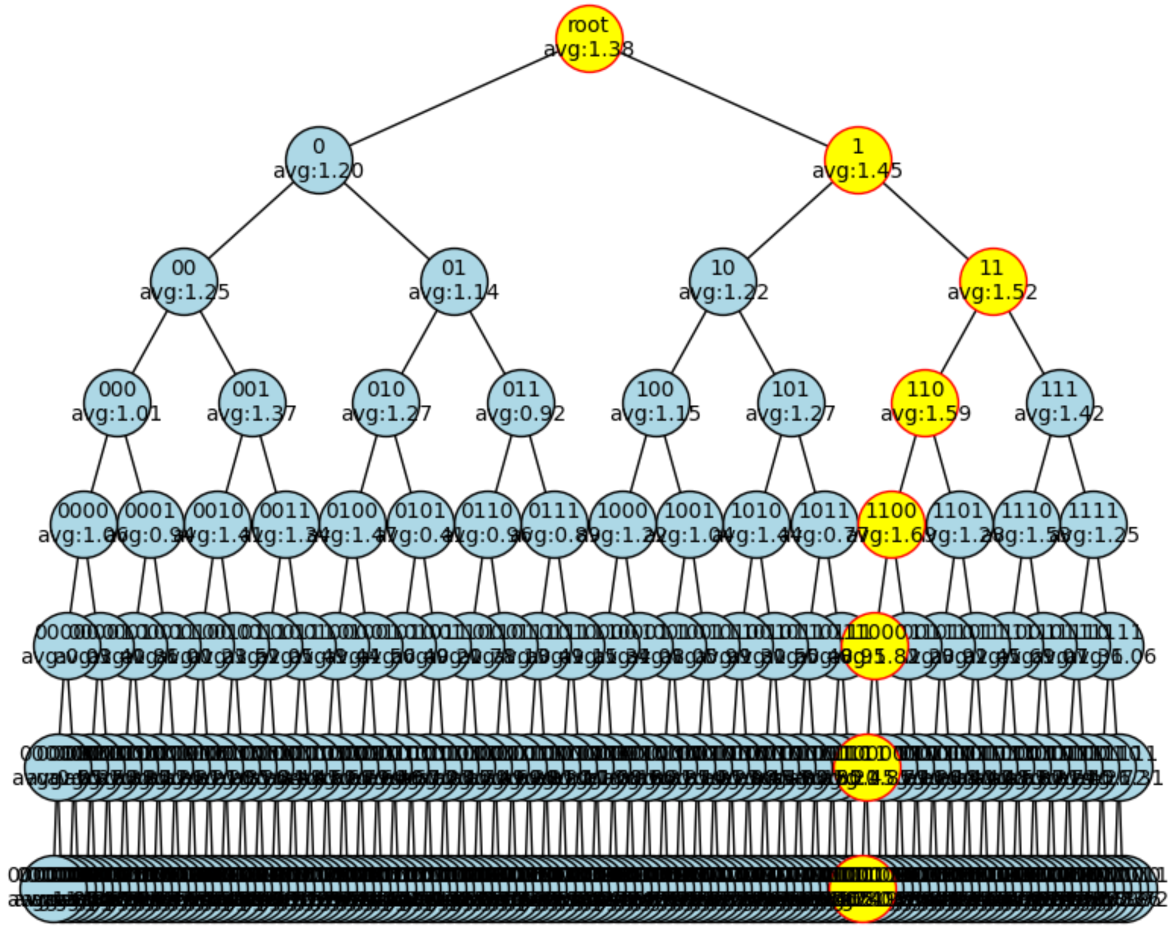



```
Iteration 1500:
MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000 -> 110001 ->
1100010
MCTS Optimal Reward: 1.800
```

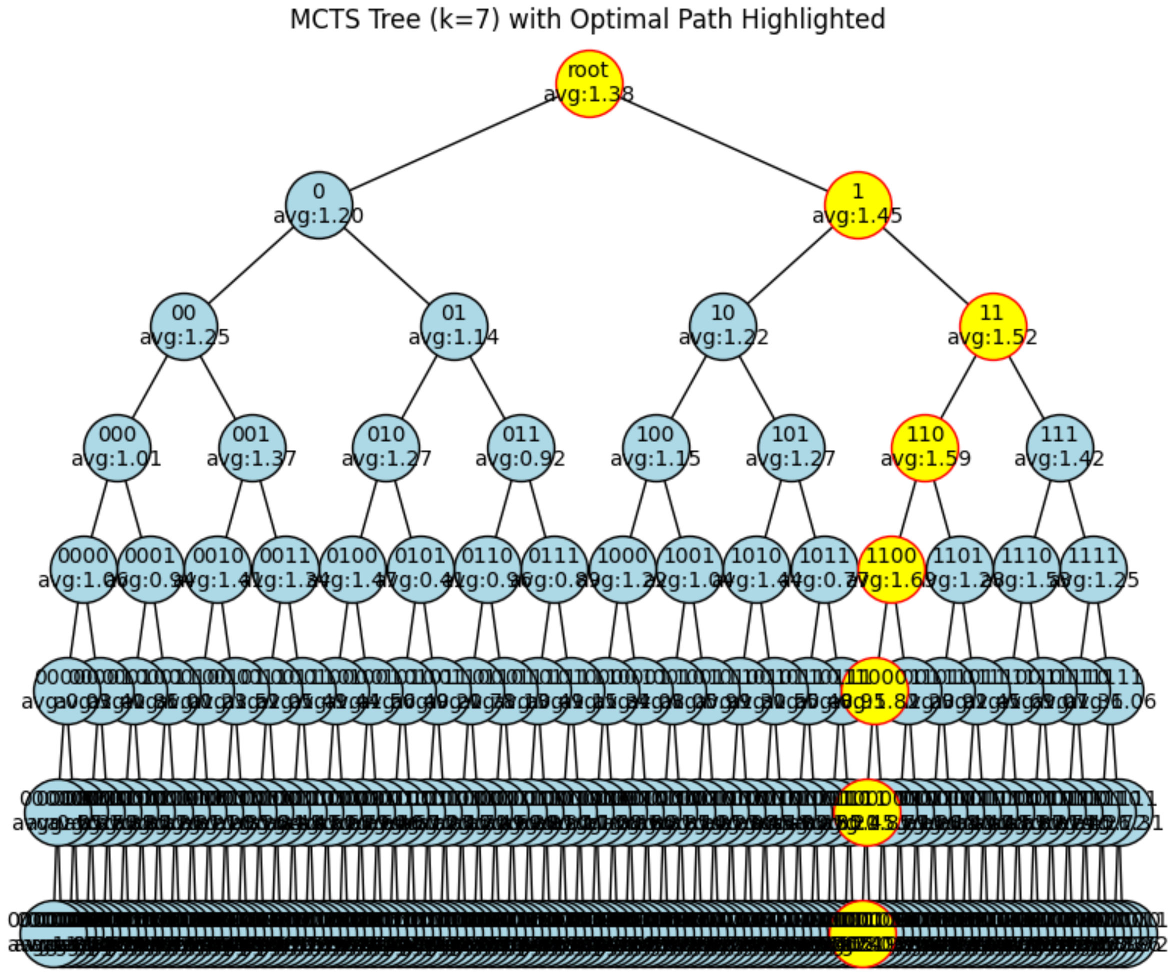



```
Iteration 2000:
MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000 -> 110000 ->
1100000
MCTS Optimal Reward: 2.023
```

MCTS Tree (k=7) with Optimal Path Highlighted



```
----- Finish MCTS for k = 7 -----

----- Print the final results -----

MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000 -> 110000 ->
1100000
MCTS Optimal Reward: 2.023
```




```
True Optimal Path: root -> 1 -> 10 -> 101 -> 1010 -> 10101 -> 101011 ->
1010110
True Optimal Reward: 2.295
MCTS Optimal Path: root -> 1 -> 11 -> 110 -> 1100 -> 11000 -> 110000 ->
1100000
MCTS Optimal Reward: 2.023
```

In [ ]:
```python
random.seed(42)

# --- True Reward Tree for k = 12 ---

K = 12
print(f"Building the True Reward Tree for k = {K}...")
global_terminal_rewards.clear()  # Clear previous rewards if any.
global_terminal_rewards.update(generate_terminal_rewards(K, gap=0.5))

# Print the reward for each terminal state.
print(f"\nTerminal Rewards for k = {K}:")
num_terminals = 2**K
print(f"\nWhen k = {K}, there are {num_terminals} actions.")
for i in range(num_terminals):
    state = format(i, f"0{K}b")
    print(f"State: {state} - Reward: {global_terminal_rewards[state]:.

# Build and plot the full true reward tree.
```

```
true_tree = build_full_tree(K)
true_opt_path, true_opt_reward = find_true_optimal_path(true_tree)
print("\nTrue Optimal Path:", " -> ".join(true_opt_path))
print(f"True Optimal Reward: {true_opt_reward:.3f}")
plot_true_tree(true_tree, K, true_opt_path)

# --- MCTS for k = 10 ---
print(f"\nRunning MCTS for k = {K}...")
mcts_root = MCTS_Node("")
iterations = 20000
plot_every = 5000
mcts(mcts_root, iterations=iterations, plot_every=plot_every, k_value=
mcts_opt_path, mcts_opt_reward = find_mcts_optimal_path(mcts_root)
print("\nMCTS Optimal Path:", " -> ".join(mcts_opt_path))
print(f"MCTS Optimal Reward: {mcts_opt_reward:.3f}")
mcts_plot_tree(mcts_root, K)

# --- Structured Results ---
print(f"\n----- Structured Results for k = {K} -----")
print("True Optimal Path:", " -> ".join(true_opt_path))
print("True Optimal Reward:", f"{true_opt_reward:.3f}")
print("MCTS Optimal Path:", " -> ".join(mcts_opt_path))
print("MCTS Optimal Reward:", f"{mcts_opt_reward:.3f}")
```

```
Building the True Reward Tree for k = 12...

Terminal Rewards for k = 12:

When k = 12, there are 4096 actions.
State: 000000000000 - Reward: 1.28
State: 000000000001 - Reward: 0.05
State: 000000000010 - Reward: 0.55
State: 000000000011 - Reward: 0.45
State: 000000000100 - Reward: 1.47
State: 000000000101 - Reward: 1.35
State: 000000000110 - Reward: 1.78
State: 000000000111 - Reward: 0.17
State: 000000001000 - Reward: 0.84
State: 000000001001 - Reward: 0.06
State: 000000001010 - Reward: 0.44
State: 000000001011 - Reward: 1.01
State: 000000001100 - Reward: 0.05
State: 000000001101 - Reward: 0.40
State: 000000001110 - Reward: 1.30
State: 000000001111 - Reward: 1.09
State: 000000010000 - Reward: 0.44
State: 000000010001 - Reward: 1.18
State: 000000010010 - Reward: 1.62
State: 000000010011 - Reward: 0.01
State: 000000010100 - Reward: 1.61
State: 000000010101 - Reward: 1.40
State: 000000010110 - Reward: 0.68
State: 000000010111 - Reward: 0.31
```

```
State: 000000011000 - Reward: 1.91
State: 000000011001 - Reward: 0.67
State: 000000011010 - Reward: 0.19
State: 000000011011 - Reward: 0.19
State: 000000011100 - Reward: 1.69
State: 000000011101 - Reward: 1.21
State: 000000011110 - Reward: 1.61
State: 000000011111 - Reward: 1.46
State: 000000100000 - Reward: 1.07
State: 000000100001 - Reward: 1.95
State: 000000100010 - Reward: 0.76
State: 000000100011 - Reward: 1.10
State: 000000100100 - Reward: 1.66
State: 000000100101 - Reward: 1.24
State: 000000100110 - Reward: 1.72
State: 000000100111 - Reward: 1.15
State: 000000101000 - Reward: 1.41
State: 000000101001 - Reward: 0.09
State: 000000101010 - Reward: 0.46
State: 000000101011 - Reward: 0.58
State: 000000101100 - Reward: 0.16
State: 000000101101 - Reward: 0.47
State: 000000101110 - Reward: 0.20
State: 000000101111 - Reward: 0.56
State: 000000110000 - Reward: 1.27
State: 000000110001 - Reward: 0.73
State: 000000110010 - Reward: 0.74
State: 000000110011 - Reward: 0.42
State: 000000110100 - Reward: 0.53
State: 000000110101 - Reward: 1.87
State: 000000110110 - Reward: 1.30
State: 000000110111 - Reward: 1.22
State: 000000111000 - Reward: 0.34
State: 000000111001 - Reward: 1.46
State: 000000111010 - Reward: 0.33
State: 000000111011 - Reward: 0.76
State: 000000111100 - Reward: 1.98
State: 000000111101 - Reward: 1.28
State: 000000111110 - Reward: 1.11
State: 000000111111 - Reward: 1.37
State: 000001000000 - Reward: 1.69
State: 000001000001 - Reward: 1.55
State: 000001000010 - Reward: 0.46
State: 000001000011 - Reward: 0.06
State: 000001000100 - Reward: 0.63
State: 000001000101 - Reward: 0.54
State: 000001000110 - Reward: 0.42
State: 000001000111 - Reward: 1.89
State: 000001001000 - Reward: 1.75
State: 000001001001 - Reward: 0.63
State: 000001001010 - Reward: 1.31
State: 000001001011 - Reward: 0.79
```

```
State: 000001001100 - Reward: 1.83
State: 000001001101 - Reward: 0.92
State: 000001001110 - Reward: 0.53
State: 000001001111 - Reward: 0.49
State: 000001010000 - Reward: 1.12
State: 000001010001 - Reward: 0.53
State: 000001010010 - Reward: 1.17
State: 000001010011 - Reward: 1.80
State: 000001010100 - Reward: 0.80
State: 000001010101 - Reward: 0.44
State: 000001010110 - Reward: 2.00
State: 000001010111 - Reward: 1.02
State: 000001011000 - Reward: 0.18
State: 000001011001 - Reward: 0.09
State: 000001011010 - Reward: 0.22
State: 000001011011 - Reward: 1.25
State: 000001011100 - Reward: 1.58
State: 000001011101 - Reward: 0.84
State: 000001011110 - Reward: 0.13
State: 000001011111 - Reward: 0.76
State: 000001100000 - Reward: 1.99
State: 000001100001 - Reward: 1.06
State: 000001100010 - Reward: 1.94
State: 000001100011 - Reward: 1.72
State: 000001100100 - Reward: 0.02
State: 000001100101 - Reward: 1.44
State: 000001100110 - Reward: 1.36
State: 000001100111 - Reward: 1.07
State: 000001101000 - Reward: 0.53
State: 000001101001 - Reward: 1.28
State: 000001101010 - Reward: 0.22
State: 000001101011 - Reward: 0.87
State: 000001101100 - Reward: 0.91
State: 000001101101 - Reward: 1.91
State: 000001101110 - Reward: 1.75
State: 000001101111 - Reward: 0.53
State: 000001110000 - Reward: 1.00
State: 000001110001 - Reward: 0.36
State: 000001110010 - Reward: 1.83
State: 000001110011 - Reward: 1.74
State: 000001110100 - Reward: 0.60
State: 000001110101 - Reward: 1.28
State: 000001110110 - Reward: 1.22
State: 000001110111 - Reward: 0.31
State: 000001111000 - Reward: 1.53
State: 000001111001 - Reward: 1.08
State: 000001111010 - Reward: 1.56
State: 000001111011 - Reward: 1.06
State: 000001111100 - Reward: 0.00
State: 000001111101 - Reward: 0.65
State: 000001111110 - Reward: 0.04
State: 000001111111 - Reward: 1.86
```

```
State: 000010000000 - Reward: 1.76
State: 000010000001 - Reward: 1.66
State: 000010000010 - Reward: 0.62
State: 000010000011 - Reward: 0.12
State: 000010000100 - Reward: 1.76
State: 000010000101 - Reward: 1.89
State: 000010000110 - Reward: 0.17
State: 000010000111 - Reward: 0.97
State: 000010001000 - Reward: 0.14
State: 000010001001 - Reward: 1.52
State: 000010001010 - Reward: 1.53
State: 000010001011 - Reward: 0.26
State: 000010001100 - Reward: 0.95
State: 000010001101 - Reward: 1.10
State: 000010001110 - Reward: 0.53
State: 000010001111 - Reward: 1.74
State: 000010010000 - Reward: 0.85
State: 000010010001 - Reward: 0.42
State: 000010010010 - Reward: 1.08
State: 000010010011 - Reward: 1.46
State: 000010010100 - Reward: 0.40
State: 000010010101 - Reward: 0.62
State: 000010010110 - Reward: 1.99
State: 000010010111 - Reward: 1.30
State: 000010011000 - Reward: 0.88
State: 000010011001 - Reward: 1.04
State: 000010011010 - Reward: 0.24
State: 000010011011 - Reward: 0.45
State: 000010011100 - Reward: 0.68
State: 000010011101 - Reward: 1.18
State: 000010011110 - Reward: 0.46
State: 000010011111 - Reward: 0.44
State: 000010100000 - Reward: 0.14
State: 000010100001 - Reward: 1.26
State: 000010100010 - Reward: 0.46
State: 000010100011 - Reward: 1.81
State: 000010100100 - Reward: 1.72
State: 000010100101 - Reward: 0.14
State: 000010100110 - Reward: 0.48
State: 000010100111 - Reward: 1.34
State: 000010101000 - Reward: 0.43
State: 000010101001 - Reward: 0.26
State: 000010101010 - Reward: 1.87
State: 000010101011 - Reward: 1.14
State: 000010101100 - Reward: 0.95
State: 000010101101 - Reward: 1.57
State: 000010101110 - Reward: 1.61
State: 000010101111 - Reward: 0.38
State: 000010110000 - Reward: 0.19
State: 000010110001 - Reward: 0.86
State: 000010110010 - Reward: 0.85
State: 000010110011 - Reward: 0.93
```

```
State: 000010110100 - Reward: 1.46
State: 000010110101 - Reward: 1.35
State: 000010110110 - Reward: 1.97
State: 000010110111 - Reward: 0.20
State: 000010111000 - Reward: 0.81
State: 000010111001 - Reward: 0.68
State: 000010111010 - Reward: 1.72
State: 000010111011 - Reward: 0.50
State: 000010111100 - Reward: 0.38
State: 000010111101 - Reward: 0.90
State: 000010111110 - Reward: 0.84
State: 000010111111 - Reward: 0.56
State: 000011000000 - Reward: 0.50
State: 000011000001 - Reward: 1.85
State: 000011000010 - Reward: 0.89
State: 000011000011 - Reward: 1.72
State: 000011000100 - Reward: 1.10
State: 000011000101 - Reward: 0.10
State: 000011000110 - Reward: 2.00
State: 000011000111 - Reward: 1.67
State: 000011001000 - Reward: 1.94
State: 000011001001 - Reward: 1.85
State: 000011001010 - Reward: 1.70
State: 000011001011 - Reward: 0.33
State: 000011001100 - Reward: 0.97
State: 000011001101 - Reward: 0.43
State: 000011001110 - Reward: 0.80
State: 000011001111 - Reward: 0.12
State: 000011010000 - Reward: 0.76
State: 000011010001 - Reward: 1.97
State: 000011010010 - Reward: 0.53
State: 000011010011 - Reward: 1.57
State: 000011010100 - Reward: 0.91
State: 000011010101 - Reward: 0.85
State: 000011010110 - Reward: 1.91
State: 000011010111 - Reward: 1.99
State: 000011011000 - Reward: 1.11
State: 000011011001 - Reward: 1.44
State: 000011011010 - Reward: 0.31
State: 000011011011 - Reward: 0.59
State: 000011011100 - Reward: 1.94
State: 000011011101 - Reward: 1.16
State: 000011011110 - Reward: 1.08
State: 000011011111 - Reward: 1.50
State: 000011100000 - Reward: 0.11
State: 000011100001 - Reward: 1.17
State: 000011100010 - Reward: 1.01
State: 000011100011 - Reward: 1.71
State: 000011100100 - Reward: 0.31
State: 000011100101 - Reward: 1.92
State: 000011100110 - Reward: 0.16
State: 000011100111 - Reward: 0.37
```

```
State: 000011101000 - Reward: 1.19
State: 000011101001 - Reward: 1.35
State: 000011101010 - Reward: 0.47
State: 000011101011 - Reward: 0.24
State: 000011101100 - Reward: 1.78
State: 000011101101 - Reward: 0.49
State: 000011101110 - Reward: 1.19
State: 000011101111 - Reward: 1.24
State: 000011110000 - Reward: 0.84
State: 000011110001 - Reward: 1.17
State: 000011110010 - Reward: 1.05
State: 000011110011 - Reward: 1.87
State: 000011110100 - Reward: 0.41
State: 000011110101 - Reward: 1.43
State: 000011110110 - Reward: 0.48
State: 000011110111 - Reward: 0.79
State: 000011111000 - Reward: 1.34
State: 000011111001 - Reward: 0.60
State: 000011111010 - Reward: 0.63
State: 000011111011 - Reward: 1.50
State: 000011111100 - Reward: 0.15
State: 000011111101 - Reward: 0.92
State: 000011111110 - Reward: 2.00
State: 000011111111 - Reward: 1.99
State: 000100000000 - Reward: 0.15
State: 000100000001 - Reward: 0.43
State: 000100000010 - Reward: 0.53
State: 000100000011 - Reward: 1.87
State: 000100000100 - Reward: 1.76
State: 000100000101 - Reward: 1.76
State: 000100000110 - Reward: 0.74
State: 000100000111 - Reward: 0.32
State: 000100001000 - Reward: 1.67
State: 000100001001 - Reward: 1.41
State: 000100001010 - Reward: 1.22
State: 000100001011 - Reward: 1.97
State: 000100001100 - Reward: 1.31
State: 000100001101 - Reward: 0.02
State: 000100001110 - Reward: 1.63
State: 000100001111 - Reward: 0.60
State: 000100010000 - Reward: 1.33
State: 000100010001 - Reward: 1.88
State: 000100010010 - Reward: 0.27
State: 000100010011 - Reward: 0.23
State: 000100010100 - Reward: 0.21
State: 000100010101 - Reward: 1.11
State: 000100010110 - Reward: 0.54
State: 000100010111 - Reward: 1.21
State: 000100011000 - Reward: 1.44
State: 000100011001 - Reward: 0.41
State: 000100011010 - Reward: 1.27
State: 000100011011 - Reward: 0.53
```

```
State: 000100011100 - Reward: 0.98
State: 000100011101 - Reward: 1.81
State: 000100011110 - Reward: 1.69
State: 000100011111 - Reward: 0.18
State: 000100100000 - Reward: 0.85
State: 000100100001 - Reward: 0.55
State: 000100100010 - Reward: 0.01
State: 000100100011 - Reward: 1.54
State: 000100100100 - Reward: 1.27
State: 000100100101 - Reward: 0.52
State: 000100100110 - Reward: 1.48
State: 000100100111 - Reward: 1.10
State: 000100101000 - Reward: 0.86
State: 000100101001 - Reward: 0.02
State: 000100101010 - Reward: 0.15
State: 000100101011 - Reward: 1.77
State: 000100101100 - Reward: 1.81
State: 000100101101 - Reward: 1.09
State: 000100101110 - Reward: 1.67
State: 000100101111 - Reward: 1.17
State: 000100110000 - Reward: 0.30
State: 000100110001 - Reward: 0.25
State: 000100110010 - Reward: 0.62
State: 000100110011 - Reward: 1.80
State: 000100110100 - Reward: 1.59
State: 000100110101 - Reward: 1.72
State: 000100110110 - Reward: 1.80
State: 000100110111 - Reward: 0.42
State: 000100111000 - Reward: 0.50
State: 000100111001 - Reward: 0.21
State: 000100111010 - Reward: 1.56
State: 000100111011 - Reward: 1.77
State: 000100111100 - Reward: 0.81
State: 000100111101 - Reward: 1.24
State: 000100111110 - Reward: 0.31
State: 000100111111 - Reward: 1.86
State: 000101000000 - Reward: 1.73
State: 000101000001 - Reward: 1.95
State: 000101000010 - Reward: 1.62
State: 000101000011 - Reward: 1.76
State: 000101000100 - Reward: 0.05
State: 000101000101 - Reward: 1.47
State: 000101000110 - Reward: 0.66
State: 000101000111 - Reward: 1.86
State: 000101001000 - Reward: 1.60
State: 000101001001 - Reward: 1.73
State: 000101001010 - Reward: 1.62
State: 000101001011 - Reward: 0.53
State: 000101001100 - Reward: 1.57
State: 000101001101 - Reward: 0.22
State: 000101001110 - Reward: 1.74
State: 000101001111 - Reward: 1.72
```

```
State: 000101010000 - Reward: 0.44
State: 000101010001 - Reward: 1.63
State: 000101010010 - Reward: 0.92
State: 000101010011 - Reward: 0.61
State: 000101010100 - Reward: 1.59
State: 000101010101 - Reward: 0.46
State: 000101010110 - Reward: 0.05
State: 000101010111 - Reward: 0.39
State: 000101011000 - Reward: 0.66
State: 000101011001 - Reward: 1.73
State: 000101011010 - Reward: 1.93
State: 000101011011 - Reward: 0.56
State: 000101011100 - Reward: 1.28
State: 000101011101 - Reward: 0.80
State: 000101011110 - Reward: 1.96
State: 000101011111 - Reward: 1.07
State: 000101100000 - Reward: 1.88
State: 000101100001 - Reward: 0.23
State: 000101100010 - Reward: 1.94
State: 000101100011 - Reward: 0.36
State: 000101100100 - Reward: 1.93
State: 000101100101 - Reward: 0.53
State: 000101100110 - Reward: 0.22
State: 000101100111 - Reward: 0.87
State: 000101101000 - Reward: 1.46
State: 000101101001 - Reward: 0.63
State: 000101101010 - Reward: 1.21
State: 000101101011 - Reward: 1.02
State: 000101101100 - Reward: 0.77
State: 000101101101 - Reward: 1.15
State: 000101101110 - Reward: 0.51
State: 000101101111 - Reward: 1.42
State: 000101110000 - Reward: 0.00
State: 000101110001 - Reward: 1.85
State: 000101110010 - Reward: 1.08
State: 000101110011 - Reward: 1.44
State: 000101110100 - Reward: 1.48
State: 000101110101 - Reward: 1.34
State: 000101110110 - Reward: 0.73
State: 000101110111 - Reward: 0.14
State: 000101111000 - Reward: 1.33
State: 000101111001 - Reward: 0.66
State: 000101111010 - Reward: 0.63
State: 000101111011 - Reward: 1.70
State: 000101111100 - Reward: 1.44
State: 000101111101 - Reward: 0.60
State: 000101111110 - Reward: 0.62
State: 000101111111 - Reward: 0.82
State: 000110000000 - Reward: 0.80
State: 000110000001 - Reward: 0.59
State: 000110000010 - Reward: 0.25
State: 000110000011 - Reward: 0.84
```

```
State: 000110000100 - Reward: 1.88
State: 000110000101 - Reward: 1.35
State: 000110000110 - Reward: 1.81
State: 000110000111 - Reward: 1.23
State: 000110001000 - Reward: 0.60
State: 000110001001 - Reward: 1.10
State: 000110001010 - Reward: 0.00
State: 000110001011 - Reward: 0.57
State: 000110001100 - Reward: 0.86
State: 000110001101 - Reward: 1.16
State: 000110001110 - Reward: 1.31
State: 000110001111 - Reward: 0.93
State: 000110010000 - Reward: 0.88
State: 000110010001 - Reward: 0.43
State: 000110010010 - Reward: 0.95
State: 000110010011 - Reward: 1.80
State: 000110010100 - Reward: 1.59
State: 000110010101 - Reward: 0.34
State: 000110010110 - Reward: 0.17
State: 000110010111 - Reward: 1.03
State: 000110011000 - Reward: 1.27
State: 000110011001 - Reward: 0.67
State: 000110011010 - Reward: 1.64
State: 000110011011 - Reward: 1.50
State: 000110011100 - Reward: 1.35
State: 000110011101 - Reward: 0.45
State: 000110011110 - Reward: 0.40
State: 000110011111 - Reward: 0.05
State: 000110100000 - Reward: 0.49
State: 000110100001 - Reward: 0.95
State: 000110100010 - Reward: 1.70
State: 000110100011 - Reward: 0.15
State: 000110100100 - Reward: 0.83
State: 000110100101 - Reward: 1.26
State: 000110100110 - Reward: 0.39
State: 000110100111 - Reward: 1.39
State: 000110101000 - Reward: 0.99
State: 000110101001 - Reward: 0.49
State: 000110101010 - Reward: 1.31
State: 000110101011 - Reward: 0.01
State: 000110101100 - Reward: 1.50
State: 000110101101 - Reward: 1.54
State: 000110101110 - Reward: 0.21
State: 000110101111 - Reward: 0.85
State: 000110110000 - Reward: 0.35
State: 000110110001 - Reward: 1.92
State: 000110110010 - Reward: 1.04
State: 000110110011 - Reward: 0.10
State: 000110110100 - Reward: 0.50
State: 000110110101 - Reward: 1.70
State: 000110110110 - Reward: 0.91
State: 000110110111 - Reward: 1.60
```

```
State: 000110111000 - Reward: 1.34
State: 000110111001 - Reward: 1.98
State: 000110111010 - Reward: 1.19
State: 000110111011 - Reward: 1.90
State: 000110111100 - Reward: 1.78
State: 000110111101 - Reward: 1.23
State: 000110111110 - Reward: 1.44
State: 000110111111 - Reward: 1.01
State: 000111000000 - Reward: 1.66
State: 000111000001 - Reward: 1.10
State: 000111000010 - Reward: 1.79
State: 000111000011 - Reward: 1.49
State: 000111000100 - Reward: 0.95
State: 000111000101 - Reward: 0.52
State: 000111000110 - Reward: 0.49
State: 000111000111 - Reward: 1.28
State: 000111001000 - Reward: 1.53
State: 000111001001 - Reward: 1.04
State: 000111001010 - Reward: 1.25
State: 000111001011 - Reward: 0.55
State: 000111001100 - Reward: 0.15
State: 000111001101 - Reward: 0.57
State: 000111001110 - Reward: 0.54
State: 000111001111 - Reward: 0.64
State: 000111010000 - Reward: 1.08
State: 000111010001 - Reward: 0.28
State: 000111010010 - Reward: 0.46
State: 000111010011 - Reward: 1.39
State: 000111010100 - Reward: 1.41
State: 000111010101 - Reward: 0.13
State: 000111010110 - Reward: 0.82
State: 000111010111 - Reward: 1.09
State: 000111011000 - Reward: 0.83
State: 000111011001 - Reward: 0.41
State: 000111011010 - Reward: 0.84
State: 000111011011 - Reward: 1.81
State: 000111011100 - Reward: 1.17
State: 000111011101 - Reward: 1.39
State: 000111011110 - Reward: 1.71
State: 000111011111 - Reward: 1.53
State: 000111100000 - Reward: 0.76
State: 000111100001 - Reward: 0.01
State: 000111100010 - Reward: 0.70
State: 000111100011 - Reward: 1.51
State: 000111100100 - Reward: 1.71
State: 000111100101 - Reward: 1.91
State: 000111100110 - Reward: 0.84
State: 000111100111 - Reward: 1.50
State: 000111101000 - Reward: 1.09
State: 000111101001 - Reward: 1.21
State: 000111101010 - Reward: 0.44
State: 000111101011 - Reward: 0.44
```

```
State: 000111101100 - Reward: 0.87
State: 000111101101 - Reward: 0.06
State: 000111101110 - Reward: 0.67
State: 000111101111 - Reward: 1.36
State: 000111110000 - Reward: 0.81
State: 000111110001 - Reward: 0.33
State: 000111110010 - Reward: 0.93
State: 000111110011 - Reward: 0.26
State: 000111110100 - Reward: 1.24
State: 000111110101 - Reward: 0.05
State: 000111110110 - Reward: 0.79
State: 000111110111 - Reward: 1.13
State: 000111111000 - Reward: 0.05
State: 000111111001 - Reward: 1.29
State: 000111111010 - Reward: 0.27
State: 000111111011 - Reward: 0.92
State: 000111111100 - Reward: 0.10
State: 000111111101 - Reward: 0.76
State: 000111111110 - Reward: 0.42
State: 000111111111 - Reward: 0.65
State: 001000000000 - Reward: 1.52
State: 001000000001 - Reward: 0.76
State: 001000000010 - Reward: 1.50
State: 001000000011 - Reward: 1.66
State: 001000000100 - Reward: 0.50
State: 001000000101 - Reward: 0.16
State: 001000000110 - Reward: 0.04
State: 001000000111 - Reward: 1.08
State: 001000001000 - Reward: 2.50
State: 001000001001 - Reward: 0.70
State: 001000001010 - Reward: 1.30
State: 001000001011 - Reward: 1.56
State: 001000001100 - Reward: 1.30
State: 001000001101 - Reward: 1.51
State: 001000001110 - Reward: 1.90
State: 001000001111 - Reward: 0.40
State: 001000010000 - Reward: 0.04
State: 001000010001 - Reward: 0.30
State: 001000010010 - Reward: 0.25
State: 001000010011 - Reward: 1.34
State: 001000010100 - Reward: 1.13
State: 001000010101 - Reward: 0.44
State: 001000010110 - Reward: 1.40
State: 001000010111 - Reward: 1.53
State: 001000011000 - Reward: 0.34
State: 001000011001 - Reward: 1.21
State: 001000011010 - Reward: 1.50
State: 001000011011 - Reward: 0.23
State: 001000011100 - Reward: 1.64
State: 001000011101 - Reward: 1.93
State: 001000011110 - Reward: 0.22
State: 001000011111 - Reward: 0.05
```

```
State: 001000100000 - Reward: 0.62
State: 001000100001 - Reward: 1.35
State: 001000100010 - Reward: 1.92
State: 001000100011 - Reward: 0.79
State: 001000100100 - Reward: 1.43
State: 001000100101 - Reward: 0.15
State: 001000100110 - Reward: 1.38
State: 001000100111 - Reward: 1.25
State: 001000101000 - Reward: 0.20
State: 001000101001 - Reward: 1.54
State: 001000101010 - Reward: 1.70
State: 001000101011 - Reward: 1.20
State: 001000101100 - Reward: 0.24
State: 001000101101 - Reward: 1.97
State: 001000101110 - Reward: 1.57
State: 001000101111 - Reward: 0.69
State: 001000110000 - Reward: 0.86
State: 001000110001 - Reward: 0.74
State: 001000110010 - Reward: 1.01
State: 001000110011 - Reward: 0.68
State: 001000110100 - Reward: 1.70
State: 001000110101 - Reward: 1.64
State: 001000110110 - Reward: 0.21
State: 001000110111 - Reward: 1.92
State: 001000111000 - Reward: 1.27
State: 001000111001 - Reward: 1.66
State: 001000111010 - Reward: 1.41
State: 001000111011 - Reward: 0.87
State: 001000111100 - Reward: 1.47
State: 001000111101 - Reward: 1.93
State: 001000111110 - Reward: 0.54
State: 001000111111 - Reward: 1.62
State: 001001000000 - Reward: 1.08
State: 001001000001 - Reward: 0.97
State: 001001000010 - Reward: 0.87
State: 001001000011 - Reward: 1.46
State: 001001000100 - Reward: 0.54
State: 001001000101 - Reward: 1.70
State: 001001000110 - Reward: 1.66
State: 001001000111 - Reward: 0.17
State: 001001001000 - Reward: 1.76
State: 001001001001 - Reward: 0.49
State: 001001001010 - Reward: 0.93
State: 001001001011 - Reward: 1.22
State: 001001001100 - Reward: 0.76
State: 001001001101 - Reward: 0.06
State: 001001001110 - Reward: 1.70
State: 001001001111 - Reward: 0.36
State: 001001010000 - Reward: 0.42
State: 001001010001 - Reward: 1.60
State: 001001010010 - Reward: 0.68
State: 001001010011 - Reward: 1.76
```

```
State: 001001010100 - Reward: 1.40
State: 001001010101 - Reward: 0.55
State: 001001010110 - Reward: 0.02
State: 001001010111 - Reward: 1.90
State: 001001011000 - Reward: 0.17
State: 001001011001 - Reward: 1.44
State: 001001011010 - Reward: 0.98
State: 001001011011 - Reward: 1.52
State: 001001011100 - Reward: 1.38
State: 001001011101 - Reward: 1.29
State: 001001011110 - Reward: 0.98
State: 001001011111 - Reward: 1.59
State: 001001100000 - Reward: 0.19
State: 001001100001 - Reward: 0.44
State: 001001100010 - Reward: 1.38
State: 001001100011 - Reward: 0.61
State: 001001100100 - Reward: 1.16
State: 001001100101 - Reward: 0.95
State: 001001100110 - Reward: 1.06
State: 001001100111 - Reward: 0.85
State: 001001101000 - Reward: 1.49
State: 001001101001 - Reward: 0.66
State: 001001101010 - Reward: 1.41
State: 001001101011 - Reward: 0.54
State: 001001101100 - Reward: 0.50
State: 001001101101 - Reward: 0.24
State: 001001101110 - Reward: 0.39
State: 001001101111 - Reward: 0.24
State: 001001110000 - Reward: 1.07
State: 001001110001 - Reward: 1.52
State: 001001110010 - Reward: 0.37
State: 001001110011 - Reward: 0.43
State: 001001110100 - Reward: 0.97
State: 001001110101 - Reward: 1.45
State: 001001110110 - Reward: 1.95
State: 001001110111 - Reward: 1.05
State: 001001111000 - Reward: 0.57
State: 001001111001 - Reward: 0.20
State: 001001111010 - Reward: 0.39
State: 001001111011 - Reward: 0.45
State: 001001111100 - Reward: 0.36
State: 001001111101 - Reward: 0.03
State: 001001111110 - Reward: 1.07
State: 001001111111 - Reward: 0.55
State: 001010000000 - Reward: 1.95
State: 001010000001 - Reward: 1.11
State: 001010000010 - Reward: 1.39
State: 001010000011 - Reward: 0.25
State: 001010000100 - Reward: 1.74
State: 001010000101 - Reward: 0.98
State: 001010000110 - Reward: 1.75
State: 001010000111 - Reward: 1.15
```

```
State: 001010001000 - Reward: 0.94
State: 001010001001 - Reward: 0.88
State: 001010001010 - Reward: 0.37
State: 001010001011 - Reward: 0.10
State: 001010001100 - Reward: 1.88
State: 001010001101 - Reward: 0.96
State: 001010001110 - Reward: 1.64
State: 001010001111 - Reward: 0.80
State: 001010010000 - Reward: 0.15
State: 001010010001 - Reward: 1.26
State: 001010010010 - Reward: 0.11
State: 001010010011 - Reward: 0.30
State: 001010010100 - Reward: 1.13
State: 001010010101 - Reward: 0.61
State: 001010010110 - Reward: 1.99
State: 001010010111 - Reward: 0.24
State: 001010011000 - Reward: 1.53
State: 001010011001 - Reward: 1.21
State: 001010011010 - Reward: 1.58
State: 001010011011 - Reward: 0.45
State: 001010011100 - Reward: 1.05
State: 001010011101 - Reward: 0.90
State: 001010011110 - Reward: 0.89
State: 001010011111 - Reward: 1.72
State: 001010100000 - Reward: 1.98
State: 001010100001 - Reward: 0.61
State: 001010100010 - Reward: 1.24
State: 001010100011 - Reward: 1.22
State: 001010100100 - Reward: 1.48
State: 001010100101 - Reward: 1.90
State: 001010100110 - Reward: 0.42
State: 001010100111 - Reward: 0.42
State: 001010101000 - Reward: 1.32
State: 001010101001 - Reward: 0.31
State: 001010101010 - Reward: 0.35
State: 001010101011 - Reward: 0.15
State: 001010101100 - Reward: 0.01
State: 001010101101 - Reward: 0.90
State: 001010101110 - Reward: 1.19
State: 001010101111 - Reward: 0.58
State: 001010110000 - Reward: 0.46
State: 001010110001 - Reward: 1.41
State: 001010110010 - Reward: 1.41
State: 001010110011 - Reward: 0.91
State: 001010110100 - Reward: 1.37
State: 001010110101 - Reward: 1.85
State: 001010110110 - Reward: 1.58
State: 001010110111 - Reward: 1.25
State: 001010111000 - Reward: 1.32
State: 001010111001 - Reward: 1.87
State: 001010111010 - Reward: 0.85
State: 001010111011 - Reward: 1.09
```

```
State: 001010111100 - Reward: 1.30
State: 001010111101 - Reward: 1.82
State: 001010111110 - Reward: 1.65
State: 001010111111 - Reward: 0.14
State: 001011000000 - Reward: 0.33
State: 001011000001 - Reward: 0.62
State: 001011000010 - Reward: 1.50
State: 001011000011 - Reward: 1.14
State: 001011000100 - Reward: 0.58
State: 001011000101 - Reward: 0.25
State: 001011000110 - Reward: 1.38
State: 001011000111 - Reward: 1.40
State: 001011001000 - Reward: 1.89
State: 001011001001 - Reward: 1.00
State: 001011001010 - Reward: 0.99
State: 001011001011 - Reward: 0.16
State: 001011001100 - Reward: 0.08
State: 001011001101 - Reward: 0.86
State: 001011001110 - Reward: 0.64
State: 001011001111 - Reward: 0.50
State: 001011010000 - Reward: 0.18
State: 001011010001 - Reward: 1.92
State: 001011010010 - Reward: 1.67
State: 001011010011 - Reward: 1.15
State: 001011010100 - Reward: 1.90
State: 001011010101 - Reward: 2.00
State: 001011010110 - Reward: 1.34
State: 001011010111 - Reward: 0.54
State: 001011011000 - Reward: 0.08
State: 001011011001 - Reward: 1.51
State: 001011011010 - Reward: 0.94
State: 001011011011 - Reward: 1.30
State: 001011011100 - Reward: 1.83
State: 001011011101 - Reward: 0.36
State: 001011011110 - Reward: 1.17
State: 001011011111 - Reward: 1.27
State: 001011100000 - Reward: 0.98
State: 001011100001 - Reward: 0.18
State: 001011100010 - Reward: 0.70
State: 001011100011 - Reward: 0.67
State: 001011100100 - Reward: 1.34
State: 001011100101 - Reward: 1.72
State: 001011100110 - Reward: 0.66
State: 001011100111 - Reward: 1.39
State: 001011101000 - Reward: 0.58
State: 001011101001 - Reward: 1.89
State: 001011101010 - Reward: 1.63
State: 001011101011 - Reward: 1.10
State: 001011101100 - Reward: 0.91
State: 001011101101 - Reward: 0.63
State: 001011101110 - Reward: 0.65
State: 001011101111 - Reward: 1.94
```

```
State: 001011110000 - Reward: 0.81
State: 001011110001 - Reward: 1.03
State: 001011110010 - Reward: 1.98
State: 001011110011 - Reward: 1.32
State: 001011110100 - Reward: 1.09
State: 001011110101 - Reward: 0.83
State: 001011110110 - Reward: 0.38
State: 001011110111 - Reward: 0.72
State: 001011111000 - Reward: 1.51
State: 001011111001 - Reward: 1.25
State: 001011111010 - Reward: 1.52
State: 001011111011 - Reward: 0.41
State: 001011111100 - Reward: 1.10
State: 001011111101 - Reward: 1.86
State: 001011111110 - Reward: 0.88
State: 001011111111 - Reward: 1.40
State: 001100000000 - Reward: 0.24
State: 001100000001 - Reward: 1.95
State: 001100000010 - Reward: 1.22
State: 001100000011 - Reward: 0.48
State: 001100000100 - Reward: 0.32
State: 001100000101 - Reward: 1.10
State: 001100000110 - Reward: 1.10
State: 001100000111 - Reward: 0.19
State: 001100001000 - Reward: 1.98
State: 001100001001 - Reward: 1.83
State: 001100001010 - Reward: 0.92
State: 001100001011 - Reward: 0.23
State: 001100001100 - Reward: 1.66
State: 001100001101 - Reward: 1.00
State: 001100001110 - Reward: 1.43
State: 001100001111 - Reward: 1.02
State: 001100010000 - Reward: 0.55
State: 001100010001 - Reward: 1.67
State: 001100010010 - Reward: 1.96
State: 001100010011 - Reward: 0.49
State: 001100010100 - Reward: 1.10
State: 001100010101 - Reward: 0.77
State: 001100010110 - Reward: 1.84
State: 001100010111 - Reward: 1.02
State: 001100011000 - Reward: 1.76
State: 001100011001 - Reward: 1.73
State: 001100011010 - Reward: 0.55
State: 001100011011 - Reward: 1.58
State: 001100011100 - Reward: 0.83
State: 001100011101 - Reward: 1.87
State: 001100011110 - Reward: 1.02
State: 001100011111 - Reward: 1.64
State: 001100100000 - Reward: 0.57
State: 001100100001 - Reward: 0.60
State: 001100100010 - Reward: 1.17
State: 001100100011 - Reward: 2.00
```

```
State: 001100100100 - Reward: 0.98
State: 001100100101 - Reward: 0.30
State: 001100100110 - Reward: 1.08
State: 001100100111 - Reward: 0.69
State: 001100101000 - Reward: 1.10
State: 001100101001 - Reward: 1.09
State: 001100101010 - Reward: 0.91
State: 001100101011 - Reward: 0.64
State: 001100101100 - Reward: 0.38
State: 001100101101 - Reward: 1.39
State: 001100101110 - Reward: 1.14
State: 001100101111 - Reward: 0.47
State: 001100110000 - Reward: 1.55
State: 001100110001 - Reward: 0.09
State: 001100110010 - Reward: 1.49
State: 001100110011 - Reward: 1.41
State: 001100110100 - Reward: 1.62
State: 001100110101 - Reward: 0.77
State: 001100110110 - Reward: 1.33
State: 001100110111 - Reward: 1.64
State: 001100111000 - Reward: 1.96
State: 001100111001 - Reward: 0.99
State: 001100111010 - Reward: 0.07
State: 001100111011 - Reward: 1.00
State: 001100111100 - Reward: 1.18
State: 001100111101 - Reward: 1.74
State: 001100111110 - Reward: 1.75
State: 001100111111 - Reward: 0.88
State: 001101000000 - Reward: 1.05
State: 001101000001 - Reward: 0.91
State: 001101000010 - Reward: 1.44
State: 001101000011 - Reward: 0.82
State: 001101000100 - Reward: 1.31
State: 001101000101 - Reward: 0.31
State: 001101000110 - Reward: 0.94
State: 001101000111 - Reward: 1.94
State: 001101001000 - Reward: 0.68
State: 001101001001 - Reward: 1.39
State: 001101001010 - Reward: 1.30
State: 001101001011 - Reward: 1.70
State: 001101001100 - Reward: 1.70
State: 001101001101 - Reward: 1.72
State: 001101001110 - Reward: 0.76
State: 001101001111 - Reward: 0.63
State: 001101010000 - Reward: 1.44
State: 001101010001 - Reward: 1.52
State: 001101010010 - Reward: 1.74
State: 001101010011 - Reward: 0.07
State: 001101010100 - Reward: 0.14
State: 001101010101 - Reward: 1.26
State: 001101010110 - Reward: 1.84
State: 001101010111 - Reward: 1.99
```

```
State: 001101011000 - Reward: 1.49
State: 001101011001 - Reward: 0.87
State: 001101011010 - Reward: 0.20
State: 001101011011 - Reward: 1.27
State: 001101011100 - Reward: 1.75
State: 001101011101 - Reward: 0.89
State: 001101011110 - Reward: 1.39
State: 001101011111 - Reward: 1.81
State: 001101100000 - Reward: 0.09
State: 001101100001 - Reward: 1.59
State: 001101100010 - Reward: 0.59
State: 001101100011 - Reward: 0.75
State: 001101100100 - Reward: 0.29
State: 001101100101 - Reward: 1.06
State: 001101100110 - Reward: 1.13
State: 001101100111 - Reward: 1.59
State: 001101101000 - Reward: 0.34
State: 001101101001 - Reward: 0.16
State: 001101101010 - Reward: 1.74
State: 001101101011 - Reward: 1.24
State: 001101101100 - Reward: 0.48
State: 001101101101 - Reward: 1.83
State: 001101101110 - Reward: 0.29
State: 001101101111 - Reward: 0.92
State: 001101110000 - Reward: 0.51
State: 001101110001 - Reward: 0.51
State: 001101110010 - Reward: 0.02
State: 001101110011 - Reward: 1.61
State: 001101110100 - Reward: 1.80
State: 001101110101 - Reward: 1.36
State: 001101110110 - Reward: 0.32
State: 001101110111 - Reward: 0.88
State: 001101111000 - Reward: 0.69
State: 001101111001 - Reward: 1.18
State: 001101111010 - Reward: 1.28
State: 001101111011 - Reward: 0.85
State: 001101111100 - Reward: 0.50
State: 001101111101 - Reward: 1.69
State: 001101111110 - Reward: 0.40
State: 001101111111 - Reward: 0.77
State: 001110000000 - Reward: 0.97
State: 001110000001 - Reward: 0.47
State: 001110000010 - Reward: 1.14
State: 001110000011 - Reward: 1.15
State: 001110000100 - Reward: 1.99
State: 001110000101 - Reward: 0.59
State: 001110000110 - Reward: 1.96
State: 001110000111 - Reward: 1.32
State: 001110001000 - Reward: 0.55
State: 001110001001 - Reward: 1.13
State: 001110001010 - Reward: 1.37
State: 001110001011 - Reward: 1.49
```

```
State: 001110001100 - Reward: 0.10
State: 001110001101 - Reward: 1.21
State: 001110001110 - Reward: 0.99
State: 001110001111 - Reward: 1.81
State: 001110010000 - Reward: 0.57
State: 001110010001 - Reward: 1.60
State: 001110010010 - Reward: 1.21
State: 001110010011 - Reward: 0.70
State: 001110010100 - Reward: 1.27
State: 001110010101 - Reward: 1.24
State: 001110010110 - Reward: 1.36
State: 001110010111 - Reward: 1.44
State: 001110011000 - Reward: 1.32
State: 001110011001 - Reward: 1.68
State: 001110011010 - Reward: 1.26
State: 001110011011 - Reward: 1.81
State: 001110011100 - Reward: 1.29
State: 001110011101 - Reward: 0.62
State: 001110011110 - Reward: 0.88
State: 001110011111 - Reward: 1.16
State: 001110100000 - Reward: 1.46
State: 001110100001 - Reward: 0.18
State: 001110100010 - Reward: 0.59
State: 001110100011 - Reward: 1.49
State: 001110100100 - Reward: 0.35
State: 001110100101 - Reward: 0.26
State: 001110100110 - Reward: 1.08
State: 001110100111 - Reward: 1.94
State: 001110101000 - Reward: 1.06
State: 001110101001 - Reward: 1.83
State: 001110101010 - Reward: 1.66
State: 001110101011 - Reward: 0.51
State: 001110101100 - Reward: 1.65
State: 001110101101 - Reward: 0.96
State: 001110101110 - Reward: 1.61
State: 001110101111 - Reward: 1.49
State: 001110110000 - Reward: 0.68
State: 001110110001 - Reward: 0.23
State: 001110110010 - Reward: 1.93
State: 001110110011 - Reward: 0.28
State: 001110110100 - Reward: 1.93
State: 001110110101 - Reward: 1.72
State: 001110110110 - Reward: 1.45
State: 001110110111 - Reward: 1.96
State: 001110111000 - Reward: 1.93
State: 001110111001 - Reward: 1.61
State: 001110111010 - Reward: 0.73
State: 001110111011 - Reward: 1.58
State: 001110111100 - Reward: 0.03
State: 001110111101 - Reward: 1.07
State: 001110111110 - Reward: 0.91
State: 001110111111 - Reward: 1.35
```

```
State: 001111000000 - Reward: 1.34
State: 001111000001 - Reward: 1.17
State: 001111000010 - Reward: 1.64
State: 001111000011 - Reward: 1.88
State: 001111000100 - Reward: 0.22
State: 001111000101 - Reward: 0.47
State: 001111000110 - Reward: 0.05
State: 001111000111 - Reward: 1.77
State: 001111001000 - Reward: 1.12
State: 001111001001 - Reward: 1.83
State: 001111001010 - Reward: 0.44
State: 001111001011 - Reward: 0.13
State: 001111001100 - Reward: 1.65
State: 001111001101 - Reward: 1.82
State: 001111001110 - Reward: 0.60
State: 001111001111 - Reward: 0.82
State: 001111010000 - Reward: 0.28
State: 001111010001 - Reward: 1.89
State: 001111010010 - Reward: 0.61
State: 001111010011 - Reward: 0.99
State: 001111010100 - Reward: 0.19
State: 001111010101 - Reward: 1.77
State: 001111010110 - Reward: 0.27
State: 001111010111 - Reward: 0.91
State: 001111011000 - Reward: 1.34
State: 001111011001 - Reward: 1.49
State: 001111011010 - Reward: 1.89
State: 001111011011 - Reward: 0.84
State: 001111011100 - Reward: 1.48
State: 001111011101 - Reward: 0.31
State: 001111011110 - Reward: 0.83
State: 001111011111 - Reward: 0.20
State: 001111100000 - Reward: 0.98
State: 001111100001 - Reward: 0.82
State: 001111100010 - Reward: 1.90
State: 001111100011 - Reward: 0.07
State: 001111100100 - Reward: 0.74
State: 001111100101 - Reward: 0.89
State: 001111100110 - Reward: 1.90
State: 001111100111 - Reward: 1.71
State: 001111101000 - Reward: 0.20
State: 001111101001 - Reward: 1.37
State: 001111101010 - Reward: 1.09
State: 001111101011 - Reward: 1.96
State: 001111101100 - Reward: 0.72
State: 001111101101 - Reward: 0.80
State: 001111101110 - Reward: 0.38
State: 001111101111 - Reward: 0.24
State: 001111110000 - Reward: 1.70
State: 001111110001 - Reward: 0.91
State: 001111110010 - Reward: 1.33
State: 001111110011 - Reward: 1.28
```

```
State: 001111110100 - Reward: 1.19
State: 001111110101 - Reward: 0.04
State: 001111110110 - Reward: 1.57
State: 001111110111 - Reward: 0.49
State: 001111111000 - Reward: 0.25
State: 001111111001 - Reward: 1.13
State: 001111111010 - Reward: 0.14
State: 001111111011 - Reward: 1.53
State: 001111111100 - Reward: 0.41
State: 001111111101 - Reward: 0.43
State: 001111111110 - Reward: 1.74
State: 001111111111 - Reward: 0.66
State: 010000000000 - Reward: 0.30
State: 010000000001 - Reward: 1.80
State: 010000000010 - Reward: 0.01
State: 010000000011 - Reward: 1.72
State: 010000000100 - Reward: 0.29
State: 010000000101 - Reward: 0.26
State: 010000000110 - Reward: 0.50
State: 010000000111 - Reward: 0.35
State: 010000001000 - Reward: 1.32
State: 010000001001 - Reward: 0.05
State: 010000001010 - Reward: 0.03
State: 010000001011 - Reward: 1.58
State: 010000001100 - Reward: 0.48
State: 010000001101 - Reward: 0.65
State: 010000001110 - Reward: 0.35
State: 010000001111 - Reward: 0.10
State: 010000010000 - Reward: 1.48
State: 010000010001 - Reward: 1.05
State: 010000010010 - Reward: 1.49
State: 010000010011 - Reward: 0.95
State: 010000010100 - Reward: 1.56
State: 010000010101 - Reward: 1.03
State: 010000010110 - Reward: 0.22
State: 010000010111 - Reward: 1.01
State: 010000011000 - Reward: 1.89
State: 010000011001 - Reward: 0.09
State: 010000011010 - Reward: 1.57
State: 010000011011 - Reward: 1.73
State: 010000011100 - Reward: 1.04
State: 010000011101 - Reward: 0.92
State: 010000011110 - Reward: 1.93
State: 010000011111 - Reward: 0.12
State: 010000100000 - Reward: 0.96
State: 010000100001 - Reward: 0.80
State: 010000100010 - Reward: 1.37
State: 010000100011 - Reward: 0.98
State: 010000100100 - Reward: 1.82
State: 010000100101 - Reward: 0.15
State: 010000100110 - Reward: 0.16
State: 010000100111 - Reward: 1.22
```

```
State: 010000101000 - Reward: 0.13
State: 010000101001 - Reward: 0.55
State: 010000101010 - Reward: 1.27
State: 010000101011 - Reward: 1.10
State: 010000101100 - Reward: 0.65
State: 010000101101 - Reward: 1.99
State: 010000101110 - Reward: 1.06
State: 010000101111 - Reward: 0.91
State: 010000110000 - Reward: 1.21
State: 010000110001 - Reward: 0.20
State: 010000110010 - Reward: 1.40
State: 010000110011 - Reward: 1.71
State: 010000110100 - Reward: 1.30
State: 010000110101 - Reward: 1.54
State: 010000110110 - Reward: 1.44
State: 010000110111 - Reward: 0.43
State: 010000111000 - Reward: 0.90
State: 010000111001 - Reward: 0.46
State: 010000111010 - Reward: 0.68
State: 010000111011 - Reward: 0.91
State: 010000111100 - Reward: 0.83
State: 010000111101 - Reward: 0.19
State: 010000111110 - Reward: 0.85
State: 010000111111 - Reward: 1.33
State: 010001000000 - Reward: 0.75
State: 010001000001 - Reward: 0.31
State: 010001000010 - Reward: 1.85
State: 010001000011 - Reward: 0.13
State: 010001000100 - Reward: 1.66
State: 010001000101 - Reward: 0.19
State: 010001000110 - Reward: 0.19
State: 010001000111 - Reward: 1.48
State: 010001001000 - Reward: 1.62
State: 010001001001 - Reward: 1.11
State: 010001001010 - Reward: 1.17
State: 010001001011 - Reward: 1.12
State: 010001001100 - Reward: 0.66
State: 010001001101 - Reward: 0.24
State: 010001001110 - Reward: 0.71
State: 010001001111 - Reward: 1.33
State: 010001010000 - Reward: 1.50
State: 010001010001 - Reward: 1.74
State: 010001010010 - Reward: 1.44
State: 010001010011 - Reward: 1.94
State: 010001010100 - Reward: 1.20
State: 010001010101 - Reward: 0.70
State: 010001010110 - Reward: 1.16
State: 010001010111 - Reward: 0.43
State: 010001011000 - Reward: 1.31
State: 010001011001 - Reward: 0.45
State: 010001011010 - Reward: 0.22
State: 010001011011 - Reward: 1.69
```

```
State: 010001011100 - Reward: 0.74
State: 010001011101 - Reward: 1.53
State: 010001011110 - Reward: 1.15
State: 010001011111 - Reward: 1.61
State: 010001100000 - Reward: 1.69
State: 010001100001 - Reward: 1.95
State: 010001100010 - Reward: 1.64
State: 010001100011 - Reward: 1.23
State: 010001100100 - Reward: 1.29
State: 010001100101 - Reward: 0.05
State: 010001100110 - Reward: 1.86
State: 010001100111 - Reward: 1.66
State: 010001101000 - Reward: 0.53
State: 010001101001 - Reward: 0.36
State: 010001101010 - Reward: 1.41
State: 010001101011 - Reward: 0.62
State: 010001101100 - Reward: 0.68
State: 010001101101 - Reward: 0.01
State: 010001101110 - Reward: 1.74
State: 010001101111 - Reward: 1.13
State: 010001110000 - Reward: 0.80
State: 010001110001 - Reward: 0.28
State: 010001110010 - Reward: 1.27
State: 010001110011 - Reward: 0.06
State: 010001110100 - Reward: 1.49
State: 010001110101 - Reward: 0.43
State: 010001110110 - Reward: 0.84
State: 010001110111 - Reward: 0.68
State: 010001111000 - Reward: 0.74
State: 010001111001 - Reward: 1.44
State: 010001111010 - Reward: 1.55
State: 010001111011 - Reward: 1.14
State: 010001111100 - Reward: 0.17
State: 010001111101 - Reward: 0.11
State: 010001111110 - Reward: 0.31
State: 010001111111 - Reward: 1.24
State: 010010000000 - Reward: 1.35
State: 010010000001 - Reward: 0.54
State: 010010000010 - Reward: 1.32
State: 010010000011 - Reward: 0.97
State: 010010000100 - Reward: 0.88
State: 010010000101 - Reward: 0.55
State: 010010000110 - Reward: 1.51
State: 010010000111 - Reward: 0.23
State: 010010001000 - Reward: 0.86
State: 010010001001 - Reward: 0.57
State: 010010001010 - Reward: 1.36
State: 010010001011 - Reward: 0.97
State: 010010001100 - Reward: 1.33
State: 010010001101 - Reward: 0.09
State: 010010001110 - Reward: 0.79
State: 010010001111 - Reward: 1.20
```

```
State: 010010010000 - Reward: 0.02
State: 010010010001 - Reward: 0.60
State: 010010010010 - Reward: 0.42
State: 010010010011 - Reward: 0.27
State: 010010010100 - Reward: 0.51
State: 010010010101 - Reward: 0.66
State: 010010010110 - Reward: 0.02
State: 010010010111 - Reward: 1.49
State: 010010011000 - Reward: 0.35
State: 010010011001 - Reward: 0.76
State: 010010011010 - Reward: 1.41
State: 010010011011 - Reward: 1.00
State: 010010011100 - Reward: 1.67
State: 010010011101 - Reward: 1.61
State: 010010011110 - Reward: 0.14
State: 010010011111 - Reward: 1.72
State: 010010100000 - Reward: 0.08
State: 010010100001 - Reward: 0.04
State: 010010100010 - Reward: 1.84
State: 010010100011 - Reward: 1.72
State: 010010100100 - Reward: 1.15
State: 010010100101 - Reward: 1.15
State: 010010100110 - Reward: 1.42
State: 010010100111 - Reward: 0.84
State: 010010101000 - Reward: 0.23
State: 010010101001 - Reward: 0.04
State: 010010101010 - Reward: 0.65
State: 010010101011 - Reward: 1.60
State: 010010101100 - Reward: 1.24
State: 010010101101 - Reward: 1.66
State: 010010101110 - Reward: 1.84
State: 010010101111 - Reward: 0.18
State: 010010110000 - Reward: 1.69
State: 010010110001 - Reward: 0.49
State: 010010110010 - Reward: 1.18
State: 010010110011 - Reward: 1.05
State: 010010110100 - Reward: 0.79
State: 010010110101 - Reward: 0.62
State: 010010110110 - Reward: 0.68
State: 010010110111 - Reward: 0.67
State: 010010111000 - Reward: 0.34
State: 010010111001 - Reward: 1.02
State: 010010111010 - Reward: 0.23
State: 010010111011 - Reward: 1.02
State: 010010111100 - Reward: 1.81
State: 010010111101 - Reward: 0.70
State: 010010111110 - Reward: 1.45
State: 010010111111 - Reward: 1.64
State: 010011000000 - Reward: 1.63
State: 010011000001 - Reward: 0.47
State: 010011000010 - Reward: 0.29
State: 010011000011 - Reward: 0.39
```

```
State: 010011000100 - Reward: 1.20
State: 010011000101 - Reward: 1.52
State: 010011000110 - Reward: 1.31
State: 010011000111 - Reward: 0.35
State: 010011001000 - Reward: 1.55
State: 010011001001 - Reward: 0.99
State: 010011001010 - Reward: 1.51
State: 010011001011 - Reward: 1.52
State: 010011001100 - Reward: 0.90
State: 010011001101 - Reward: 1.85
State: 010011001110 - Reward: 1.13
State: 010011001111 - Reward: 1.27
State: 010011010000 - Reward: 1.25
State: 010011010001 - Reward: 1.73
State: 010011010010 - Reward: 1.25
State: 010011010011 - Reward: 0.30
State: 010011010100 - Reward: 0.14
State: 010011010101 - Reward: 0.88
State: 010011010110 - Reward: 0.61
State: 010011010111 - Reward: 0.55
State: 010011011000 - Reward: 0.11
State: 010011011001 - Reward: 1.01
State: 010011011010 - Reward: 0.62
State: 010011011011 - Reward: 0.90
State: 010011011100 - Reward: 0.11
State: 010011011101 - Reward: 1.66
State: 010011011110 - Reward: 0.15
State: 010011011111 - Reward: 1.73
State: 010011100000 - Reward: 1.71
State: 010011100001 - Reward: 1.23
State: 010011100010 - Reward: 1.01
State: 010011100011 - Reward: 0.93
State: 010011100100 - Reward: 1.11
State: 010011100101 - Reward: 1.58
State: 010011100110 - Reward: 1.79
State: 010011100111 - Reward: 0.90
State: 010011101000 - Reward: 1.62
State: 010011101001 - Reward: 1.30
State: 010011101010 - Reward: 0.64
State: 010011101011 - Reward: 0.95
State: 010011101100 - Reward: 0.30
State: 010011101101 - Reward: 0.12
State: 010011101110 - Reward: 0.21
State: 010011101111 - Reward: 1.80
State: 010011110000 - Reward: 0.69
State: 010011110001 - Reward: 1.43
State: 010011110010 - Reward: 1.01
State: 010011110011 - Reward: 0.35
State: 010011110100 - Reward: 0.50
State: 010011110101 - Reward: 0.88
State: 010011110110 - Reward: 0.88
State: 010011110111 - Reward: 1.05
```

```
State: 010011111000 - Reward: 0.32
State: 010011111001 - Reward: 0.75
State: 010011111010 - Reward: 0.57
State: 010011111011 - Reward: 0.82
State: 010011111100 - Reward: 0.68
State: 010011111101 - Reward: 1.20
State: 010011111110 - Reward: 1.58
State: 010011111111 - Reward: 1.29
State: 010100000000 - Reward: 0.13
State: 010100000001 - Reward: 0.19
State: 010100000010 - Reward: 1.36
State: 010100000011 - Reward: 0.57
State: 010100000100 - Reward: 1.45
State: 010100000101 - Reward: 1.31
State: 010100000110 - Reward: 1.81
State: 010100000111 - Reward: 1.75
State: 010100001000 - Reward: 0.67
State: 010100001001 - Reward: 1.17
State: 010100001010 - Reward: 0.28
State: 010100001011 - Reward: 0.70
State: 010100001100 - Reward: 1.94
State: 010100001101 - Reward: 1.40
State: 010100001110 - Reward: 0.78
State: 010100001111 - Reward: 1.19
State: 010100010000 - Reward: 1.88
State: 010100010001 - Reward: 0.62
State: 010100010010 - Reward: 0.75
State: 010100010011 - Reward: 1.58
State: 010100010100 - Reward: 1.63
State: 010100010101 - Reward: 1.34
State: 010100010110 - Reward: 1.66
State: 010100010111 - Reward: 1.48
State: 010100011000 - Reward: 1.37
State: 010100011001 - Reward: 1.05
State: 010100011010 - Reward: 1.29
State: 010100011011 - Reward: 0.85
State: 010100011100 - Reward: 0.72
State: 010100011101 - Reward: 0.73
State: 010100011110 - Reward: 0.36
State: 010100011111 - Reward: 0.43
State: 010100100000 - Reward: 1.90
State: 010100100001 - Reward: 0.97
State: 010100100010 - Reward: 0.45
State: 010100100011 - Reward: 0.28
State: 010100100100 - Reward: 0.15
State: 010100100101 - Reward: 1.69
State: 010100100110 - Reward: 0.20
State: 010100100111 - Reward: 1.54
State: 010100101000 - Reward: 1.67
State: 010100101001 - Reward: 1.77
State: 010100101010 - Reward: 0.08
State: 010100101011 - Reward: 0.67
```

```
State: 010100101100 - Reward: 1.53
State: 010100101101 - Reward: 0.26
State: 010100101110 - Reward: 0.75
State: 010100101111 - Reward: 0.32
State: 010100110000 - Reward: 1.66
State: 010100110001 - Reward: 1.54
State: 010100110010 - Reward: 1.62
State: 010100110011 - Reward: 0.33
State: 010100110100 - Reward: 0.88
State: 010100110101 - Reward: 0.82
State: 010100110110 - Reward: 1.35
State: 010100110111 - Reward: 0.48
State: 010100111000 - Reward: 0.89
State: 010100111001 - Reward: 0.57
State: 010100111010 - Reward: 1.50
State: 010100111011 - Reward: 0.90
State: 010100111100 - Reward: 1.07
State: 010100111101 - Reward: 0.62
State: 010100111110 - Reward: 1.62
State: 010100111111 - Reward: 0.94
State: 010101000000 - Reward: 1.67
State: 010101000001 - Reward: 0.74
State: 010101000010 - Reward: 1.89
State: 010101000011 - Reward: 1.97
State: 010101000100 - Reward: 0.92
State: 010101000101 - Reward: 0.56
State: 010101000110 - Reward: 0.76
State: 010101000111 - Reward: 1.05
State: 010101001000 - Reward: 1.93
State: 010101001001 - Reward: 1.63
State: 010101001010 - Reward: 1.60
State: 010101001011 - Reward: 0.28
State: 010101001100 - Reward: 0.50
State: 010101001101 - Reward: 1.28
State: 010101001110 - Reward: 1.75
State: 010101001111 - Reward: 1.11
State: 010101010000 - Reward: 0.21
State: 010101010001 - Reward: 1.69
State: 010101010010 - Reward: 1.70
State: 010101010011 - Reward: 0.57
State: 010101010100 - Reward: 1.53
State: 010101010101 - Reward: 0.55
State: 010101010110 - Reward: 1.81
State: 010101010111 - Reward: 0.29
State: 010101011000 - Reward: 0.87
State: 010101011001 - Reward: 1.89
State: 010101011010 - Reward: 0.44
State: 010101011011 - Reward: 0.90
State: 010101011100 - Reward: 0.70
State: 010101011101 - Reward: 0.05
State: 010101011110 - Reward: 0.11
State: 010101011111 - Reward: 1.00
```

```
State: 010101100000 - Reward: 0.47
State: 010101100001 - Reward: 1.99
State: 010101100010 - Reward: 0.75
State: 010101100011 - Reward: 0.06
State: 010101100100 - Reward: 1.86
State: 010101100101 - Reward: 1.68
State: 010101100110 - Reward: 1.30
State: 010101100111 - Reward: 1.58
State: 010101101000 - Reward: 0.28
State: 010101101001 - Reward: 0.57
State: 010101101010 - Reward: 1.66
State: 010101101011 - Reward: 1.39
State: 010101101100 - Reward: 0.28
State: 010101101101 - Reward: 1.41
State: 010101101110 - Reward: 0.90
State: 010101101111 - Reward: 0.01
State: 010101110000 - Reward: 0.16
State: 010101110001 - Reward: 0.51
State: 010101110010 - Reward: 1.67
State: 010101110011 - Reward: 1.10
State: 010101110100 - Reward: 1.45
State: 010101110101 - Reward: 1.06
State: 010101110110 - Reward: 0.22
State: 010101110111 - Reward: 0.58
State: 010101111000 - Reward: 0.60
State: 010101111001 - Reward: 0.10
State: 010101111010 - Reward: 0.84
State: 010101111011 - Reward: 1.59
State: 010101111100 - Reward: 0.91
State: 010101111101 - Reward: 0.22
State: 010101111110 - Reward: 1.81
State: 010101111111 - Reward: 1.19
State: 010110000000 - Reward: 0.03
State: 010110000001 - Reward: 1.03
State: 010110000010 - Reward: 0.48
State: 010110000011 - Reward: 0.29
State: 010110000100 - Reward: 0.86
State: 010110000101 - Reward: 1.23
State: 010110000110 - Reward: 0.48
State: 010110000111 - Reward: 0.83
State: 010110001000 - Reward: 1.33
State: 010110001001 - Reward: 0.17
State: 010110001010 - Reward: 1.95
State: 010110001011 - Reward: 0.14
State: 010110001100 - Reward: 1.05
State: 010110001101 - Reward: 1.01
State: 010110001110 - Reward: 1.98
State: 010110001111 - Reward: 1.11
State: 010110010000 - Reward: 0.78
State: 010110010001 - Reward: 0.94
State: 010110010010 - Reward: 1.27
State: 010110010011 - Reward: 1.96
```

```
State: 010110010100 - Reward: 0.51
State: 010110010101 - Reward: 0.03
State: 010110010110 - Reward: 1.58
State: 010110010111 - Reward: 0.69
State: 010110011000 - Reward: 1.47
State: 010110011001 - Reward: 1.26
State: 010110011010 - Reward: 1.54
State: 010110011011 - Reward: 1.47
State: 010110011100 - Reward: 0.67
State: 010110011101 - Reward: 0.09
State: 010110011110 - Reward: 1.09
State: 010110011111 - Reward: 1.63
State: 010110100000 - Reward: 0.35
State: 010110100001 - Reward: 1.56
State: 010110100010 - Reward: 0.93
State: 010110100011 - Reward: 1.39
State: 010110100100 - Reward: 1.26
State: 010110100101 - Reward: 1.62
State: 010110100110 - Reward: 0.13
State: 010110100111 - Reward: 1.55
State: 010110101000 - Reward: 0.92
State: 010110101001 - Reward: 0.59
State: 010110101010 - Reward: 0.09
State: 010110101011 - Reward: 0.40
State: 010110101100 - Reward: 0.08
State: 010110101101 - Reward: 1.87
State: 010110101110 - Reward: 1.03
State: 010110101111 - Reward: 1.98
State: 010110110000 - Reward: 1.09
State: 010110110001 - Reward: 0.51
State: 010110110010 - Reward: 1.51
State: 010110110011 - Reward: 0.38
State: 010110110100 - Reward: 0.71
State: 010110110101 - Reward: 1.56
State: 010110110110 - Reward: 1.73
State: 010110110111 - Reward: 0.66
State: 010110111000 - Reward: 0.25
State: 010110111001 - Reward: 0.74
State: 010110111010 - Reward: 1.78
State: 010110111011 - Reward: 1.49
State: 010110111100 - Reward: 1.79
State: 010110111101 - Reward: 0.77
State: 010110111110 - Reward: 1.95
State: 010110111111 - Reward: 0.99
State: 010111000000 - Reward: 1.00
State: 010111000001 - Reward: 1.85
State: 010111000010 - Reward: 1.04
State: 010111000011 - Reward: 1.60
State: 010111000100 - Reward: 1.45
State: 010111000101 - Reward: 0.16
State: 010111000110 - Reward: 1.20
State: 010111000111 - Reward: 1.64
```

```
State: 010111001000 - Reward: 1.09
State: 010111001001 - Reward: 0.64
State: 010111001010 - Reward: 0.16
State: 010111001011 - Reward: 1.32
State: 010111001100 - Reward: 0.61
State: 010111001101 - Reward: 1.21
State: 010111001110 - Reward: 0.85
State: 010111001111 - Reward: 1.38
State: 010111010000 - Reward: 0.70
State: 010111010001 - Reward: 0.08
State: 010111010010 - Reward: 1.74
State: 010111010011 - Reward: 0.71
State: 010111010100 - Reward: 2.00
State: 010111010101 - Reward: 0.55
State: 010111010110 - Reward: 1.96
State: 010111010111 - Reward: 1.90
State: 010111011000 - Reward: 0.15
State: 010111011001 - Reward: 1.28
State: 010111011010 - Reward: 0.73
State: 010111011011 - Reward: 1.60
State: 010111011100 - Reward: 1.36
State: 010111011101 - Reward: 1.91
State: 010111011110 - Reward: 0.29
State: 010111011111 - Reward: 1.22
State: 010111100000 - Reward: 1.56
State: 010111100001 - Reward: 0.07
State: 010111100010 - Reward: 0.13
State: 010111100011 - Reward: 1.56
State: 010111100100 - Reward: 0.73
State: 010111100101 - Reward: 0.77
State: 010111100110 - Reward: 1.13
State: 010111100111 - Reward: 1.21
State: 010111101000 - Reward: 1.36
State: 010111101001 - Reward: 1.90
State: 010111101010 - Reward: 0.74
State: 010111101011 - Reward: 1.53
State: 010111101100 - Reward: 1.15
State: 010111101101 - Reward: 1.06
State: 010111101110 - Reward: 0.80
State: 010111101111 - Reward: 1.30
State: 010111110000 - Reward: 0.50
State: 010111110001 - Reward: 0.23
State: 010111110010 - Reward: 1.47
State: 010111110011 - Reward: 1.00
State: 010111110100 - Reward: 0.77
State: 010111110101 - Reward: 1.12
State: 010111110110 - Reward: 0.52
State: 010111110111 - Reward: 0.52
State: 010111111000 - Reward: 0.89
State: 010111111001 - Reward: 1.99
State: 010111111010 - Reward: 0.57
State: 010111111011 - Reward: 1.83
```

```
State: 010111111100 - Reward: 0.98
State: 010111111101 - Reward: 0.25
State: 010111111110 - Reward: 1.71
State: 010111111111 - Reward: 0.90
State: 011000000000 - Reward: 1.80
State: 011000000001 - Reward: 0.89
State: 011000000010 - Reward: 0.18
State: 011000000011 - Reward: 1.36
State: 011000000100 - Reward: 1.69
State: 011000000101 - Reward: 0.64
State: 011000000110 - Reward: 0.69
State: 011000000111 - Reward: 0.13
State: 011000001000 - Reward: 1.08
State: 011000001001 - Reward: 1.78
State: 011000001010 - Reward: 1.70
State: 011000001011 - Reward: 1.42
State: 011000001100 - Reward: 1.85
State: 011000001101 - Reward: 1.28
State: 011000001110 - Reward: 1.59
State: 011000001111 - Reward: 1.02
State: 011000010000 - Reward: 0.24
State: 011000010001 - Reward: 0.40
State: 011000010010 - Reward: 0.28
State: 011000010011 - Reward: 1.58
State: 011000010100 - Reward: 0.05
State: 011000010101 - Reward: 1.11
State: 011000010110 - Reward: 0.74
State: 011000010111 - Reward: 1.61
State: 011000011000 - Reward: 1.10
State: 011000011001 - Reward: 1.22
State: 011000011010 - Reward: 0.17
State: 011000011011 - Reward: 0.62
State: 011000011100 - Reward: 2.00
State: 011000011101 - Reward: 1.44
State: 011000011110 - Reward: 1.05
State: 011000011111 - Reward: 1.54
State: 011000100000 - Reward: 1.65
State: 011000100001 - Reward: 0.15
State: 011000100010 - Reward: 1.94
State: 011000100011 - Reward: 1.28
State: 011000100100 - Reward: 0.90
State: 011000100101 - Reward: 1.36
State: 011000100110 - Reward: 0.69
State: 011000100111 - Reward: 1.76
State: 011000101000 - Reward: 1.56
State: 011000101001 - Reward: 1.28
State: 011000101010 - Reward: 0.36
State: 011000101011 - Reward: 1.93
State: 011000101100 - Reward: 0.87
State: 011000101101 - Reward: 1.82
State: 011000101110 - Reward: 0.11
State: 011000101111 - Reward: 0.25
```

```
State: 011000110000 - Reward: 0.31
State: 011000110001 - Reward: 0.33
State: 011000110010 - Reward: 0.65
State: 011000110011 - Reward: 1.42
State: 011000110100 - Reward: 0.69
State: 011000110101 - Reward: 1.88
State: 011000110110 - Reward: 1.79
State: 011000110111 - Reward: 1.69
State: 011000111000 - Reward: 0.50
State: 011000111001 - Reward: 1.27
State: 011000111010 - Reward: 1.10
State: 011000111011 - Reward: 0.25
State: 011000111100 - Reward: 0.61
State: 011000111101 - Reward: 1.07
State: 011000111110 - Reward: 1.01
State: 011000111111 - Reward: 0.34
State: 011001000000 - Reward: 1.88
State: 011001000001 - Reward: 0.31
State: 011001000010 - Reward: 1.32
State: 011001000011 - Reward: 1.44
State: 011001000100 - Reward: 1.21
State: 011001000101 - Reward: 1.69
State: 011001000110 - Reward: 1.13
State: 011001000111 - Reward: 1.65
State: 011001001000 - Reward: 0.06
State: 011001001001 - Reward: 0.09
State: 011001001010 - Reward: 1.28
State: 011001001011 - Reward: 1.15
State: 011001001100 - Reward: 1.30
State: 011001001101 - Reward: 1.53
State: 011001001110 - Reward: 0.83
State: 011001001111 - Reward: 1.28
State: 011001010000 - Reward: 1.00
State: 011001010001 - Reward: 1.25
State: 011001010010 - Reward: 0.58
State: 011001010011 - Reward: 1.91
State: 011001010100 - Reward: 0.97
State: 011001010101 - Reward: 1.61
State: 011001010110 - Reward: 1.37
State: 011001010111 - Reward: 0.59
State: 011001011000 - Reward: 0.15
State: 011001011001 - Reward: 0.12
State: 011001011010 - Reward: 0.88
State: 011001011011 - Reward: 0.97
State: 011001011100 - Reward: 0.41
State: 011001011101 - Reward: 1.21
State: 011001011110 - Reward: 0.63
State: 011001011111 - Reward: 1.44
State: 011001100000 - Reward: 1.47
State: 011001100001 - Reward: 1.72
State: 011001100010 - Reward: 1.95
State: 011001100011 - Reward: 0.26
```

```
State: 011001100100 - Reward: 0.74
State: 011001100101 - Reward: 1.12
State: 011001100110 - Reward: 0.64
State: 011001100111 - Reward: 0.93
State: 011001101000 - Reward: 0.53
State: 011001101001 - Reward: 0.50
State: 011001101010 - Reward: 0.19
State: 011001101011 - Reward: 0.58
State: 011001101100 - Reward: 0.77
State: 011001101101 - Reward: 1.23
State: 011001101110 - Reward: 0.50
State: 011001101111 - Reward: 1.73
State: 011001110000 - Reward: 0.32
State: 011001110001 - Reward: 0.65
State: 011001110010 - Reward: 1.16
State: 011001110011 - Reward: 0.63
State: 011001110100 - Reward: 1.53
State: 011001110101 - Reward: 1.00
State: 011001110110 - Reward: 1.03
State: 011001110111 - Reward: 1.00
State: 011001111000 - Reward: 0.62
State: 011001111001 - Reward: 0.05
State: 011001111010 - Reward: 1.89
State: 011001111011 - Reward: 1.01
State: 011001111100 - Reward: 1.93
State: 011001111101 - Reward: 0.43
State: 011001111110 - Reward: 0.71
State: 011001111111 - Reward: 0.10
State: 011010000000 - Reward: 0.99
State: 011010000001 - Reward: 1.76
State: 011010000010 - Reward: 1.31
State: 011010000011 - Reward: 0.94
State: 011010000100 - Reward: 1.07
State: 011010000101 - Reward: 1.69
State: 011010000110 - Reward: 0.86
State: 011010000111 - Reward: 1.76
State: 011010001000 - Reward: 1.46
State: 011010001001 - Reward: 1.53
State: 011010001010 - Reward: 0.73
State: 011010001011 - Reward: 0.80
State: 011010001100 - Reward: 1.14
State: 011010001101 - Reward: 0.39
State: 011010001110 - Reward: 1.11
State: 011010001111 - Reward: 0.15
State: 011010010000 - Reward: 1.01
State: 011010010001 - Reward: 1.53
State: 011010010010 - Reward: 0.56
State: 011010010011 - Reward: 1.98
State: 011010010100 - Reward: 1.36
State: 011010010101 - Reward: 0.24
State: 011010010110 - Reward: 1.95
State: 011010010111 - Reward: 0.79
```

```
State: 011010011000 - Reward: 1.59
State: 011010011001 - Reward: 0.68
State: 011010011010 - Reward: 1.88
State: 011010011011 - Reward: 1.51
State: 011010011100 - Reward: 0.40
State: 011010011101 - Reward: 1.02
State: 011010011110 - Reward: 1.00
State: 011010011111 - Reward: 0.09
State: 011010100000 - Reward: 0.27
State: 011010100001 - Reward: 0.67
State: 011010100010 - Reward: 0.95
State: 011010100011 - Reward: 0.91
State: 011010100100 - Reward: 1.21
State: 011010100101 - Reward: 1.03
State: 011010100110 - Reward: 0.66
State: 011010100111 - Reward: 1.23
State: 011010101000 - Reward: 0.33
State: 011010101001 - Reward: 1.98
State: 011010101010 - Reward: 1.48
State: 011010101011 - Reward: 0.60
State: 011010101100 - Reward: 0.67
State: 011010101101 - Reward: 1.66
State: 011010101110 - Reward: 1.06
State: 011010101111 - Reward: 1.42
State: 011010110000 - Reward: 0.60
State: 011010110001 - Reward: 1.63
State: 011010110010 - Reward: 0.74
State: 011010110011 - Reward: 1.35
State: 011010110100 - Reward: 1.96
State: 011010110101 - Reward: 1.17
State: 011010110110 - Reward: 1.59
State: 011010110111 - Reward: 1.45
State: 011010111000 - Reward: 1.38
State: 011010111001 - Reward: 0.05
State: 011010111010 - Reward: 0.95
State: 011010111011 - Reward: 1.93
State: 011010111100 - Reward: 1.57
State: 011010111101 - Reward: 1.55
State: 011010111110 - Reward: 1.16
State: 011010111111 - Reward: 1.44
State: 011011000000 - Reward: 1.17
State: 011011000001 - Reward: 0.34
State: 011011000010 - Reward: 1.26
State: 011011000011 - Reward: 1.24
State: 011011000100 - Reward: 1.68
State: 011011000101 - Reward: 0.30
State: 011011000110 - Reward: 1.36
State: 011011000111 - Reward: 0.06
State: 011011001000 - Reward: 1.90
State: 011011001001 - Reward: 0.22
State: 011011001010 - Reward: 0.04
State: 011011001011 - Reward: 0.63
```

```
State: 011011001100 - Reward: 0.30
State: 011011001101 - Reward: 1.38
State: 011011001110 - Reward: 0.82
State: 011011001111 - Reward: 1.55
State: 011011010000 - Reward: 1.84
State: 011011010001 - Reward: 1.75
State: 011011010010 - Reward: 1.47
State: 011011010011 - Reward: 0.12
State: 011011010100 - Reward: 0.28
State: 011011010101 - Reward: 0.41
State: 011011010110 - Reward: 0.65
State: 011011010111 - Reward: 1.32
State: 011011011000 - Reward: 1.05
State: 011011011001 - Reward: 0.63
State: 011011011010 - Reward: 0.35
State: 011011011011 - Reward: 1.82
State: 011011011100 - Reward: 0.68
State: 011011011101 - Reward: 0.71
State: 011011011110 - Reward: 1.54
State: 011011011111 - Reward: 1.44
State: 011011100000 - Reward: 1.29
State: 011011100001 - Reward: 1.39
State: 011011100010 - Reward: 1.22
State: 011011100011 - Reward: 0.38
State: 011011100100 - Reward: 0.49
State: 011011100101 - Reward: 1.12
State: 011011100110 - Reward: 0.45
State: 011011100111 - Reward: 1.95
State: 011011101000 - Reward: 0.60
State: 011011101001 - Reward: 0.58
State: 011011101010 - Reward: 0.41
State: 011011101011 - Reward: 1.41
State: 011011101100 - Reward: 0.63
State: 011011101101 - Reward: 0.70
State: 011011101110 - Reward: 1.87
State: 011011101111 - Reward: 1.59
State: 011011110000 - Reward: 0.55
State: 011011110001 - Reward: 0.24
State: 011011110010 - Reward: 1.35
State: 011011110011 - Reward: 0.76
State: 011011110100 - Reward: 1.96
State: 011011110101 - Reward: 1.64
State: 011011110110 - Reward: 1.91
State: 011011110111 - Reward: 1.61
State: 011011111000 - Reward: 0.58
State: 011011111001 - Reward: 0.58
State: 011011111010 - Reward: 1.43
State: 011011111011 - Reward: 0.69
State: 011011111100 - Reward: 0.88
State: 011011111101 - Reward: 0.51
State: 011011111110 - Reward: 0.96
State: 011011111111 - Reward: 0.40
```

```
State: 011100000000 - Reward: 1.08
State: 011100000001 - Reward: 1.87
State: 011100000010 - Reward: 1.39
State: 011100000011 - Reward: 0.27
State: 011100000100 - Reward: 1.23
State: 011100000101 - Reward: 1.17
State: 011100000110 - Reward: 0.48
State: 011100000111 - Reward: 1.34
State: 011100001000 - Reward: 1.06
State: 011100001001 - Reward: 1.28
State: 011100001010 - Reward: 0.10
State: 011100001011 - Reward: 0.83
State: 011100001100 - Reward: 1.43
State: 011100001101 - Reward: 0.20
State: 011100001110 - Reward: 1.54
State: 011100001111 - Reward: 0.01
State: 011100010000 - Reward: 1.10
State: 011100010001 - Reward: 1.86
State: 011100010010 - Reward: 0.81
State: 011100010011 - Reward: 1.87
State: 011100010100 - Reward: 1.76
State: 011100010101 - Reward: 0.95
State: 011100010110 - Reward: 0.40
State: 011100010111 - Reward: 1.93
State: 011100011000 - Reward: 0.64
State: 011100011001 - Reward: 1.29
State: 011100011010 - Reward: 1.82
State: 011100011011 - Reward: 0.18
State: 011100011100 - Reward: 1.15
State: 011100011101 - Reward: 1.07
State: 011100011110 - Reward: 1.45
State: 011100011111 - Reward: 1.87
State: 011100100000 - Reward: 1.83
State: 011100100001 - Reward: 0.35
State: 011100100010 - Reward: 1.76
State: 011100100011 - Reward: 0.35
State: 011100100100 - Reward: 1.84
State: 011100100101 - Reward: 1.99
State: 011100100110 - Reward: 0.79
State: 011100100111 - Reward: 0.99
State: 011100101000 - Reward: 1.87
State: 011100101001 - Reward: 1.92
State: 011100101010 - Reward: 1.85
State: 011100101011 - Reward: 1.75
State: 011100101100 - Reward: 0.02
State: 011100101101 - Reward: 1.14
State: 011100101110 - Reward: 0.21
State: 011100101111 - Reward: 1.97
State: 011100110000 - Reward: 0.57
State: 011100110001 - Reward: 1.98
State: 011100110010 - Reward: 1.09
State: 011100110011 - Reward: 0.99
```

```
State: 011100110100 - Reward: 1.88
State: 011100110101 - Reward: 1.70
State: 011100110110 - Reward: 0.94
State: 011100110111 - Reward: 0.39
State: 011100111000 - Reward: 0.23
State: 011100111001 - Reward: 0.32
State: 011100111010 - Reward: 0.92
State: 011100111011 - Reward: 0.51
State: 011100111100 - Reward: 0.37
State: 011100111101 - Reward: 1.47
State: 011100111110 - Reward: 1.58
State: 011100111111 - Reward: 1.14
State: 011101000000 - Reward: 1.51
State: 011101000001 - Reward: 0.35
State: 011101000010 - Reward: 1.71
State: 011101000011 - Reward: 1.79
State: 011101000100 - Reward: 1.65
State: 011101000101 - Reward: 1.03
State: 011101000110 - Reward: 0.17
State: 011101000111 - Reward: 1.34
State: 011101001000 - Reward: 0.37
State: 011101001001 - Reward: 0.28
State: 011101001010 - Reward: 0.65
State: 011101001011 - Reward: 0.50
State: 011101001100 - Reward: 0.52
State: 011101001101 - Reward: 0.47
State: 011101001110 - Reward: 1.51
State: 011101001111 - Reward: 1.91
State: 011101010000 - Reward: 0.60
State: 011101010001 - Reward: 1.45
State: 011101010010 - Reward: 0.02
State: 011101010011 - Reward: 1.31
State: 011101010100 - Reward: 1.39
State: 011101010101 - Reward: 0.12
State: 011101010110 - Reward: 0.24
State: 011101010111 - Reward: 0.61
State: 011101011000 - Reward: 0.81
State: 011101011001 - Reward: 1.01
State: 011101011010 - Reward: 1.79
State: 011101011011 - Reward: 1.41
State: 011101011100 - Reward: 0.62
State: 011101011101 - Reward: 0.23
State: 011101011110 - Reward: 1.83
State: 011101011111 - Reward: 0.59
State: 011101100000 - Reward: 1.23
State: 011101100001 - Reward: 0.44
State: 011101100010 - Reward: 0.27
State: 011101100011 - Reward: 0.31
State: 011101100100 - Reward: 1.50
State: 011101100101 - Reward: 1.21
State: 011101100110 - Reward: 0.83
State: 011101100111 - Reward: 1.10
```

```
State: 011101101000 - Reward: 0.94
State: 011101101001 - Reward: 1.08
State: 011101101010 - Reward: 1.33
State: 011101101011 - Reward: 0.44
State: 011101101100 - Reward: 0.49
State: 011101101101 - Reward: 1.51
State: 011101101110 - Reward: 1.75
State: 011101101111 - Reward: 0.16
State: 011101110000 - Reward: 0.89
State: 011101110001 - Reward: 1.41
State: 011101110010 - Reward: 0.16
State: 011101110011 - Reward: 1.13
State: 011101110100 - Reward: 0.12
State: 011101110101 - Reward: 1.10
State: 011101110110 - Reward: 1.01
State: 011101110111 - Reward: 1.15
State: 011101111000 - Reward: 0.30
State: 011101111001 - Reward: 0.66
State: 011101111010 - Reward: 1.04
State: 011101111011 - Reward: 0.23
State: 011101111100 - Reward: 0.41
State: 011101111101 - Reward: 1.17
State: 011101111110 - Reward: 0.18
State: 011101111111 - Reward: 1.02
State: 011110000000 - Reward: 1.62
State: 011110000001 - Reward: 0.91
State: 011110000010 - Reward: 1.03
State: 011110000011 - Reward: 0.91
State: 011110000100 - Reward: 0.12
State: 011110000101 - Reward: 0.92
State: 011110000110 - Reward: 1.61
State: 011110000111 - Reward: 1.45
State: 011110001000 - Reward: 0.79
State: 011110001001 - Reward: 1.63
State: 011110001010 - Reward: 1.49
State: 011110001011 - Reward: 1.16
State: 011110001100 - Reward: 0.09
State: 011110001101 - Reward: 0.69
State: 011110001110 - Reward: 0.13
State: 011110001111 - Reward: 1.99
State: 011110010000 - Reward: 1.87
State: 011110010001 - Reward: 0.14
State: 011110010010 - Reward: 1.87
State: 011110010011 - Reward: 0.06
State: 011110010100 - Reward: 0.82
State: 011110010101 - Reward: 1.54
State: 011110010110 - Reward: 1.53
State: 011110010111 - Reward: 1.96
State: 011110011000 - Reward: 1.29
State: 011110011001 - Reward: 0.84
State: 011110011010 - Reward: 1.99
State: 011110011011 - Reward: 0.76
```

```
State: 011110011100 - Reward: 1.74
State: 011110011101 - Reward: 1.81
State: 011110011110 - Reward: 0.75
State: 011110011111 - Reward: 1.37
State: 011110100000 - Reward: 1.32
State: 011110100001 - Reward: 1.08
State: 011110100010 - Reward: 1.31
State: 011110100011 - Reward: 0.70
State: 011110100100 - Reward: 0.36
State: 011110100101 - Reward: 1.07
State: 011110100110 - Reward: 1.06
State: 011110100111 - Reward: 1.46
State: 011110101000 - Reward: 0.45
State: 011110101001 - Reward: 0.01
State: 011110101010 - Reward: 0.05
State: 011110101011 - Reward: 0.60
State: 011110101100 - Reward: 1.35
State: 011110101101 - Reward: 1.09
State: 011110101110 - Reward: 1.06
State: 011110101111 - Reward: 1.65
State: 011110110000 - Reward: 0.50
State: 011110110001 - Reward: 0.69
State: 011110110010 - Reward: 0.55
State: 011110110011 - Reward: 1.87
State: 011110110100 - Reward: 1.45
State: 011110110101 - Reward: 0.23
State: 011110110110 - Reward: 1.62
State: 011110110111 - Reward: 0.84
State: 011110111000 - Reward: 1.53
State: 011110111001 - Reward: 1.77
State: 011110111010 - Reward: 0.03
State: 011110111011 - Reward: 0.41
State: 011110111100 - Reward: 0.20
State: 011110111101 - Reward: 0.07
State: 011110111110 - Reward: 1.20
State: 011110111111 - Reward: 1.41
State: 011111000000 - Reward: 0.10
State: 011111000001 - Reward: 1.48
State: 011111000010 - Reward: 0.80
State: 011111000011 - Reward: 0.47
State: 011111000100 - Reward: 0.43
State: 011111000101 - Reward: 1.73
State: 011111000110 - Reward: 0.11
State: 011111000111 - Reward: 1.01
State: 011111001000 - Reward: 0.58
State: 011111001001 - Reward: 1.63
State: 011111001010 - Reward: 1.46
State: 011111001011 - Reward: 0.64
State: 011111001100 - Reward: 1.20
State: 011111001101 - Reward: 1.35
State: 011111001110 - Reward: 0.64
State: 011111001111 - Reward: 0.60
```

```
State: 011111010000 - Reward: 0.29
State: 011111010001 - Reward: 1.32
State: 011111010010 - Reward: 0.44
State: 011111010011 - Reward: 0.60
State: 011111010100 - Reward: 0.12
State: 011111010101 - Reward: 1.90
State: 011111010110 - Reward: 1.76
State: 011111010111 - Reward: 1.82
State: 011111011000 - Reward: 1.25
State: 011111011001 - Reward: 0.85
State: 011111011010 - Reward: 0.99
State: 011111011011 - Reward: 1.94
State: 011111011100 - Reward: 1.88
State: 011111011101 - Reward: 1.34
State: 011111011110 - Reward: 1.57
State: 011111011111 - Reward: 0.64
State: 011111100000 - Reward: 0.83
State: 011111100001 - Reward: 0.30
State: 011111100010 - Reward: 0.75
State: 011111100011 - Reward: 1.51
State: 011111100100 - Reward: 0.95
State: 011111100101 - Reward: 1.70
State: 011111100110 - Reward: 0.60
State: 011111100111 - Reward: 1.42
State: 011111101000 - Reward: 1.61
State: 011111101001 - Reward: 1.83
State: 011111101010 - Reward: 1.12
State: 011111101011 - Reward: 1.94
State: 011111101100 - Reward: 1.11
State: 011111101101 - Reward: 0.27
State: 011111101110 - Reward: 0.49
State: 011111101111 - Reward: 0.41
State: 011111110000 - Reward: 1.29
State: 011111110001 - Reward: 1.84
State: 011111110010 - Reward: 1.69
State: 011111110011 - Reward: 0.18
State: 011111110100 - Reward: 1.45
State: 011111110101 - Reward: 0.38
State: 011111110110 - Reward: 0.54
State: 011111110111 - Reward: 1.35
State: 011111111000 - Reward: 1.21
State: 011111111001 - Reward: 1.75
State: 011111111010 - Reward: 0.38
State: 011111111011 - Reward: 1.52
State: 011111111100 - Reward: 1.45
State: 011111111101 - Reward: 1.12
State: 011111111110 - Reward: 0.96
State: 011111111111 - Reward: 1.74
State: 100000000000 - Reward: 0.67
State: 100000000001 - Reward: 1.91
State: 100000000010 - Reward: 0.03
State: 100000000011 - Reward: 1.87
```

```
State: 100000000100 - Reward: 1.92
State: 100000000101 - Reward: 0.23
State: 100000000110 - Reward: 2.00
State: 100000000111 - Reward: 0.96
State: 100000001000 - Reward: 0.49
State: 100000001001 - Reward: 1.21
State: 100000001010 - Reward: 0.41
State: 100000001011 - Reward: 1.83
State: 100000001100 - Reward: 1.10
State: 100000001101 - Reward: 1.55
State: 100000001110 - Reward: 0.76
State: 100000001111 - Reward: 1.07
State: 100000010000 - Reward: 0.72
State: 100000010001 - Reward: 0.52
State: 100000010010 - Reward: 1.03
State: 100000010011 - Reward: 0.99
State: 100000010100 - Reward: 0.20
State: 100000010101 - Reward: 1.96
State: 100000010110 - Reward: 0.94
State: 100000010111 - Reward: 1.68
State: 100000011000 - Reward: 1.83
State: 100000011001 - Reward: 0.74
State: 100000011010 - Reward: 0.83
State: 100000011011 - Reward: 1.13
State: 100000011100 - Reward: 0.44
State: 100000011101 - Reward: 0.29
State: 100000011110 - Reward: 0.52
State: 100000011111 - Reward: 1.87
State: 100000100000 - Reward: 1.16
State: 100000100001 - Reward: 0.84
State: 100000100010 - Reward: 0.30
State: 100000100011 - Reward: 0.66
State: 100000100100 - Reward: 0.76
State: 100000100101 - Reward: 1.67
State: 100000100110 - Reward: 1.00
State: 100000100111 - Reward: 1.31
State: 100000101000 - Reward: 1.37
State: 100000101001 - Reward: 0.51
State: 100000101010 - Reward: 1.64
State: 100000101011 - Reward: 1.93
State: 100000101100 - Reward: 1.28
State: 100000101101 - Reward: 0.98
State: 100000101110 - Reward: 0.34
State: 100000101111 - Reward: 1.59
State: 100000110000 - Reward: 0.34
State: 100000110001 - Reward: 1.44
State: 100000110010 - Reward: 0.98
State: 100000110011 - Reward: 1.83
State: 100000110100 - Reward: 1.08
State: 100000110101 - Reward: 1.28
State: 100000110110 - Reward: 0.12
State: 100000110111 - Reward: 0.07
```

```
State: 100000111000 - Reward: 1.69
State: 100000111001 - Reward: 1.89
State: 100000111010 - Reward: 1.34
State: 100000111011 - Reward: 1.53
State: 100000111100 - Reward: 0.82
State: 100000111101 - Reward: 1.69
State: 100000111110 - Reward: 0.46
State: 100000111111 - Reward: 1.41
State: 100001000000 - Reward: 0.02
State: 100001000001 - Reward: 1.01
State: 100001000010 - Reward: 0.75
State: 100001000011 - Reward: 1.24
State: 100001000100 - Reward: 1.33
State: 100001000101 - Reward: 1.23
State: 100001000110 - Reward: 0.97
State: 100001000111 - Reward: 0.98
State: 100001001000 - Reward: 0.01
State: 100001001001 - Reward: 1.10
State: 100001001010 - Reward: 0.02
State: 100001001011 - Reward: 1.06
State: 100001001100 - Reward: 0.55
State: 100001001101 - Reward: 1.95
State: 100001001110 - Reward: 0.03
State: 100001001111 - Reward: 1.63
State: 100001010000 - Reward: 1.35
State: 100001010001 - Reward: 1.61
State: 100001010010 - Reward: 1.82
State: 100001010011 - Reward: 0.21
State: 100001010100 - Reward: 0.19
State: 100001010101 - Reward: 0.30
State: 100001010110 - Reward: 0.38
State: 100001010111 - Reward: 1.05
State: 100001011000 - Reward: 1.63
State: 100001011001 - Reward: 0.53
State: 100001011010 - Reward: 0.79
State: 100001011011 - Reward: 0.75
State: 100001011100 - Reward: 0.81
State: 100001011101 - Reward: 1.13
State: 100001011110 - Reward: 1.98
State: 100001011111 - Reward: 0.45
State: 100001100000 - Reward: 1.37
State: 100001100001 - Reward: 1.70
State: 100001100010 - Reward: 1.31
State: 100001100011 - Reward: 1.72
State: 100001100100 - Reward: 1.52
State: 100001100101 - Reward: 0.19
State: 100001100110 - Reward: 0.76
State: 100001100111 - Reward: 1.11
State: 100001101000 - Reward: 0.11
State: 100001101001 - Reward: 0.02
State: 100001101010 - Reward: 0.34
State: 100001101011 - Reward: 1.00
```

```
State: 100001101100 - Reward: 0.87
State: 100001101101 - Reward: 1.57
State: 100001101110 - Reward: 1.13
State: 100001101111 - Reward: 1.72
State: 100001110000 - Reward: 0.19
State: 100001110001 - Reward: 1.06
State: 100001110010 - Reward: 0.09
State: 100001110011 - Reward: 0.42
State: 100001110100 - Reward: 1.74
State: 100001110101 - Reward: 1.78
State: 100001110110 - Reward: 0.95
State: 100001110111 - Reward: 0.09
State: 100001111000 - Reward: 0.15
State: 100001111001 - Reward: 1.85
State: 100001111010 - Reward: 1.80
State: 100001111011 - Reward: 1.13
State: 100001111100 - Reward: 0.07
State: 100001111101 - Reward: 1.86
State: 100001111110 - Reward: 0.63
State: 100001111111 - Reward: 1.92
State: 100010000000 - Reward: 1.17
State: 100010000001 - Reward: 1.50
State: 100010000010 - Reward: 1.43
State: 100010000011 - Reward: 0.80
State: 100010000100 - Reward: 0.15
State: 100010000101 - Reward: 0.32
State: 100010000110 - Reward: 0.48
State: 100010000111 - Reward: 1.67
State: 100010001000 - Reward: 0.78
State: 100010001001 - Reward: 1.79
State: 100010001010 - Reward: 0.66
State: 100010001011 - Reward: 1.51
State: 100010001100 - Reward: 0.28
State: 100010001101 - Reward: 1.98
State: 100010001110 - Reward: 1.45
State: 100010001111 - Reward: 1.00
State: 100010010000 - Reward: 1.95
State: 100010010001 - Reward: 0.11
State: 100010010010 - Reward: 0.87
State: 100010010011 - Reward: 1.68
State: 100010010100 - Reward: 0.68
State: 100010010101 - Reward: 1.54
State: 100010010110 - Reward: 1.91
State: 100010010111 - Reward: 0.79
State: 100010011000 - Reward: 1.55
State: 100010011001 - Reward: 0.06
State: 100010011010 - Reward: 0.55
State: 100010011011 - Reward: 1.99
State: 100010011100 - Reward: 0.98
State: 100010011101 - Reward: 0.71
State: 100010011110 - Reward: 1.88
State: 100010011111 - Reward: 0.86
```

```
State: 100010100000 - Reward: 1.36
State: 100010100001 - Reward: 1.32
State: 100010100010 - Reward: 0.17
State: 100010100011 - Reward: 1.24
State: 100010100100 - Reward: 1.60
State: 100010100101 - Reward: 1.43
State: 100010100110 - Reward: 0.16
State: 100010100111 - Reward: 0.31
State: 100010101000 - Reward: 1.42
State: 100010101001 - Reward: 1.27
State: 100010101010 - Reward: 1.48
State: 100010101011 - Reward: 0.63
State: 100010101100 - Reward: 0.21
State: 100010101101 - Reward: 0.01
State: 100010101110 - Reward: 0.62
State: 100010101111 - Reward: 0.72
State: 100010110000 - Reward: 0.54
State: 100010110001 - Reward: 0.27
State: 100010110010 - Reward: 0.37
State: 100010110011 - Reward: 0.90
State: 100010110100 - Reward: 1.11
State: 100010110101 - Reward: 0.82
State: 100010110110 - Reward: 0.05
State: 100010110111 - Reward: 0.71
State: 100010111000 - Reward: 0.19
State: 100010111001 - Reward: 1.20
State: 100010111010 - Reward: 0.65
State: 100010111011 - Reward: 0.77
State: 100010111100 - Reward: 0.58
State: 100010111101 - Reward: 0.78
State: 100010111110 - Reward: 0.17
State: 100010111111 - Reward: 1.80
State: 100011000000 - Reward: 1.81
State: 100011000001 - Reward: 1.96
State: 100011000010 - Reward: 1.14
State: 100011000011 - Reward: 0.34
State: 100011000100 - Reward: 0.76
State: 100011000101 - Reward: 0.28
State: 100011000110 - Reward: 0.60
State: 100011000111 - Reward: 0.99
State: 100011001000 - Reward: 0.13
State: 100011001001 - Reward: 0.87
State: 100011001010 - Reward: 0.84
State: 100011001011 - Reward: 0.97
State: 100011001100 - Reward: 0.15
State: 100011001101 - Reward: 0.50
State: 100011001110 - Reward: 0.49
State: 100011001111 - Reward: 1.25
State: 100011010000 - Reward: 1.19
State: 100011010001 - Reward: 0.39
State: 100011010010 - Reward: 0.21
State: 100011010011 - Reward: 0.61
```

```
State: 100011010100 - Reward: 1.90
State: 100011010101 - Reward: 0.66
State: 100011010110 - Reward: 1.24
State: 100011010111 - Reward: 1.61
State: 100011011000 - Reward: 0.66
State: 100011011001 - Reward: 0.67
State: 100011011010 - Reward: 1.63
State: 100011011011 - Reward: 1.72
State: 100011011100 - Reward: 1.95
State: 100011011101 - Reward: 0.27
State: 100011011110 - Reward: 0.64
State: 100011011111 - Reward: 1.89
State: 100011100000 - Reward: 0.40
State: 100011100001 - Reward: 0.63
State: 100011100010 - Reward: 1.93
State: 100011100011 - Reward: 1.94
State: 100011100100 - Reward: 0.58
State: 100011100101 - Reward: 1.39
State: 100011100110 - Reward: 0.98
State: 100011100111 - Reward: 1.15
State: 100011101000 - Reward: 0.48
State: 100011101001 - Reward: 0.75
State: 100011101010 - Reward: 1.63
State: 100011101011 - Reward: 0.79
State: 100011101100 - Reward: 0.23
State: 100011101101 - Reward: 1.13
State: 100011101110 - Reward: 1.18
State: 100011101111 - Reward: 1.09
State: 100011110000 - Reward: 1.36
State: 100011110001 - Reward: 1.10
State: 100011110010 - Reward: 1.91
State: 100011110011 - Reward: 0.92
State: 100011110100 - Reward: 1.42
State: 100011110101 - Reward: 0.88
State: 100011110110 - Reward: 0.58
State: 100011110111 - Reward: 1.39
State: 100011111000 - Reward: 1.64
State: 100011111001 - Reward: 1.59
State: 100011111010 - Reward: 0.82
State: 100011111011 - Reward: 1.00
State: 100011111100 - Reward: 1.27
State: 100011111101 - Reward: 0.48
State: 100011111110 - Reward: 1.32
State: 100011111111 - Reward: 1.43
State: 100100000000 - Reward: 1.58
State: 100100000001 - Reward: 0.15
State: 100100000010 - Reward: 1.98
State: 100100000011 - Reward: 0.96
State: 100100000100 - Reward: 0.80
State: 100100000101 - Reward: 1.01
State: 100100000110 - Reward: 1.84
State: 100100000111 - Reward: 1.38
```

```
State: 100100001000 - Reward: 1.09
State: 100100001001 - Reward: 1.58
State: 100100001010 - Reward: 0.72
State: 100100001011 - Reward: 1.79
State: 100100001100 - Reward: 1.07
State: 100100001101 - Reward: 1.28
State: 100100001110 - Reward: 0.17
State: 100100001111 - Reward: 1.54
State: 100100010000 - Reward: 1.32
State: 100100010001 - Reward: 0.71
State: 100100010010 - Reward: 1.29
State: 100100010011 - Reward: 0.09
State: 100100010100 - Reward: 1.97
State: 100100010101 - Reward: 1.35
State: 100100010110 - Reward: 0.80
State: 100100010111 - Reward: 1.51
State: 100100011000 - Reward: 1.93
State: 100100011001 - Reward: 0.86
State: 100100011010 - Reward: 0.02
State: 100100011011 - Reward: 0.52
State: 100100011100 - Reward: 1.02
State: 100100011101 - Reward: 1.04
State: 100100011110 - Reward: 1.16
State: 100100011111 - Reward: 1.15
State: 100100100000 - Reward: 0.89
State: 100100100001 - Reward: 0.78
State: 100100100010 - Reward: 1.54
State: 100100100011 - Reward: 1.18
State: 100100100100 - Reward: 1.00
State: 100100100101 - Reward: 0.69
State: 100100100110 - Reward: 0.05
State: 100100100111 - Reward: 0.21
State: 100100101000 - Reward: 0.83
State: 100100101001 - Reward: 1.92
State: 100100101010 - Reward: 0.23
State: 100100101011 - Reward: 1.88
State: 100100101100 - Reward: 0.28
State: 100100101101 - Reward: 0.62
State: 100100101110 - Reward: 0.91
State: 100100101111 - Reward: 0.41
State: 100100110000 - Reward: 0.97
State: 100100110001 - Reward: 0.95
State: 100100110010 - Reward: 0.88
State: 100100110011 - Reward: 1.39
State: 100100110100 - Reward: 0.64
State: 100100110101 - Reward: 0.60
State: 100100110110 - Reward: 1.62
State: 100100110111 - Reward: 0.23
State: 100100111000 - Reward: 1.70
State: 100100111001 - Reward: 1.30
State: 100100111010 - Reward: 1.35
State: 100100111011 - Reward: 0.33
```

```
State: 100100111100 - Reward: 1.97
State: 100100111101 - Reward: 0.49
State: 100100111110 - Reward: 0.35
State: 100100111111 - Reward: 0.32
State: 100101000000 - Reward: 1.12
State: 100101000001 - Reward: 1.92
State: 100101000010 - Reward: 0.46
State: 100101000011 - Reward: 0.81
State: 100101000100 - Reward: 0.37
State: 100101000101 - Reward: 1.28
State: 100101000110 - Reward: 0.86
State: 100101000111 - Reward: 0.06
State: 100101001000 - Reward: 1.23
State: 100101001001 - Reward: 0.39
State: 100101001010 - Reward: 1.18
State: 100101001011 - Reward: 0.78
State: 100101001100 - Reward: 1.41
State: 100101001101 - Reward: 0.41
State: 100101001110 - Reward: 1.50
State: 100101001111 - Reward: 1.62
State: 100101010000 - Reward: 0.13
State: 100101010001 - Reward: 0.20
State: 100101010010 - Reward: 1.74
State: 100101010011 - Reward: 0.37
State: 100101010100 - Reward: 0.65
State: 100101010101 - Reward: 0.92
State: 100101010110 - Reward: 0.52
State: 100101010111 - Reward: 1.73
State: 100101011000 - Reward: 1.06
State: 100101011001 - Reward: 1.28
State: 100101011010 - Reward: 1.19
State: 100101011011 - Reward: 1.22
State: 100101011100 - Reward: 1.17
State: 100101011101 - Reward: 0.70
State: 100101011110 - Reward: 1.69
State: 100101011111 - Reward: 1.23
State: 100101100000 - Reward: 1.63
State: 100101100001 - Reward: 1.41
State: 100101100010 - Reward: 0.59
State: 100101100011 - Reward: 1.23
State: 100101100100 - Reward: 0.17
State: 100101100101 - Reward: 0.27
State: 100101100110 - Reward: 0.24
State: 100101100111 - Reward: 0.61
State: 100101101000 - Reward: 0.37
State: 100101101001 - Reward: 1.39
State: 100101101010 - Reward: 1.02
State: 100101101011 - Reward: 0.84
State: 100101101100 - Reward: 0.28
State: 100101101101 - Reward: 0.77
State: 100101101110 - Reward: 0.37
State: 100101101111 - Reward: 1.27
```

```
State: 100101110000 - Reward: 1.39
State: 100101110001 - Reward: 1.29
State: 100101110010 - Reward: 2.00
State: 100101110011 - Reward: 1.11
State: 100101110100 - Reward: 0.98
State: 100101110101 - Reward: 0.28
State: 100101110110 - Reward: 0.63
State: 100101110111 - Reward: 0.90
State: 100101111000 - Reward: 0.11
State: 100101111001 - Reward: 0.72
State: 100101111010 - Reward: 0.02
State: 100101111011 - Reward: 0.27
State: 100101111100 - Reward: 1.63
State: 100101111101 - Reward: 1.93
State: 100101111110 - Reward: 1.01
State: 100101111111 - Reward: 0.99
State: 100110000000 - Reward: 1.37
State: 100110000001 - Reward: 0.83
State: 100110000010 - Reward: 1.68
State: 100110000011 - Reward: 0.98
State: 100110000100 - Reward: 0.17
State: 100110000101 - Reward: 0.06
State: 100110000110 - Reward: 1.52
State: 100110000111 - Reward: 0.58
State: 100110001000 - Reward: 0.55
State: 100110001001 - Reward: 1.08
State: 100110001010 - Reward: 0.34
State: 100110001011 - Reward: 0.91
State: 100110001100 - Reward: 1.49
State: 100110001101 - Reward: 1.53
State: 100110001110 - Reward: 1.10
State: 100110001111 - Reward: 0.23
State: 100110010000 - Reward: 0.23
State: 100110010001 - Reward: 1.55
State: 100110010010 - Reward: 1.65
State: 100110010011 - Reward: 0.73
State: 100110010100 - Reward: 1.65
State: 100110010101 - Reward: 0.08
State: 100110010110 - Reward: 1.44
State: 100110010111 - Reward: 1.09
State: 100110011000 - Reward: 1.98
State: 100110011001 - Reward: 0.20
State: 100110011010 - Reward: 1.66
State: 100110011011 - Reward: 1.50
State: 100110011100 - Reward: 0.60
State: 100110011101 - Reward: 2.00
State: 100110011110 - Reward: 0.90
State: 100110011111 - Reward: 0.70
State: 100110100000 - Reward: 1.63
State: 100110100001 - Reward: 0.88
State: 100110100010 - Reward: 1.99
State: 100110100011 - Reward: 1.55
```

```
State: 100110100100 - Reward: 0.47
State: 100110100101 - Reward: 1.62
State: 100110100110 - Reward: 1.18
State: 100110100111 - Reward: 0.70
State: 100110101000 - Reward: 1.42
State: 100110101001 - Reward: 1.27
State: 100110101010 - Reward: 0.33
State: 100110101011 - Reward: 0.28
State: 100110101100 - Reward: 0.41
State: 100110101101 - Reward: 0.41
State: 100110101110 - Reward: 0.12
State: 100110101111 - Reward: 0.70
State: 100110110000 - Reward: 0.56
State: 100110110001 - Reward: 1.08
State: 100110110010 - Reward: 0.65
State: 100110110011 - Reward: 1.41
State: 100110110100 - Reward: 0.58
State: 100110110101 - Reward: 0.53
State: 100110110110 - Reward: 1.72
State: 100110110111 - Reward: 1.97
State: 100110111000 - Reward: 1.36
State: 100110111001 - Reward: 0.19
State: 100110111010 - Reward: 1.93
State: 100110111011 - Reward: 1.57
State: 100110111100 - Reward: 1.84
State: 100110111101 - Reward: 1.98
State: 100110111110 - Reward: 1.73
State: 100110111111 - Reward: 0.25
State: 100111000000 - Reward: 1.73
State: 100111000001 - Reward: 0.50
State: 100111000010 - Reward: 1.42
State: 100111000011 - Reward: 1.66
State: 100111000100 - Reward: 1.52
State: 100111000101 - Reward: 1.35
State: 100111000110 - Reward: 0.98
State: 100111000111 - Reward: 1.15
State: 100111001000 - Reward: 0.54
State: 100111001001 - Reward: 0.83
State: 100111001010 - Reward: 0.90
State: 100111001011 - Reward: 1.27
State: 100111001100 - Reward: 1.76
State: 100111001101 - Reward: 0.19
State: 100111001110 - Reward: 1.03
State: 100111001111 - Reward: 0.56
State: 100111010000 - Reward: 1.87
State: 100111010001 - Reward: 0.74
State: 100111010010 - Reward: 1.90
State: 100111010011 - Reward: 0.65
State: 100111010100 - Reward: 0.00
State: 100111010101 - Reward: 1.55
State: 100111010110 - Reward: 1.47
State: 100111010111 - Reward: 1.46
```

```
State: 100111011000 - Reward: 0.92
State: 100111011001 - Reward: 1.33
State: 100111011010 - Reward: 0.72
State: 100111011011 - Reward: 0.13
State: 100111011100 - Reward: 1.07
State: 100111011101 - Reward: 0.44
State: 100111011110 - Reward: 0.86
State: 100111011111 - Reward: 0.42
State: 100111100000 - Reward: 0.54
State: 100111100001 - Reward: 1.66
State: 100111100010 - Reward: 0.68
State: 100111100011 - Reward: 1.16
State: 100111100100 - Reward: 1.13
State: 100111100101 - Reward: 0.97
State: 100111100110 - Reward: 0.69
State: 100111100111 - Reward: 1.37
State: 100111101000 - Reward: 0.10
State: 100111101001 - Reward: 0.20
State: 100111101010 - Reward: 1.57
State: 100111101011 - Reward: 0.92
State: 100111101100 - Reward: 0.25
State: 100111101101 - Reward: 1.72
State: 100111101110 - Reward: 0.88
State: 100111101111 - Reward: 0.00
State: 100111110000 - Reward: 1.92
State: 100111110001 - Reward: 0.40
State: 100111110010 - Reward: 1.38
State: 100111110011 - Reward: 0.26
State: 100111110100 - Reward: 1.30
State: 100111110101 - Reward: 0.32
State: 100111110110 - Reward: 1.87
State: 100111110111 - Reward: 0.55
State: 100111111000 - Reward: 1.31
State: 100111111001 - Reward: 0.50
State: 100111111010 - Reward: 0.74
State: 100111111011 - Reward: 1.81
State: 100111111100 - Reward: 0.33
State: 100111111101 - Reward: 0.79
State: 100111111110 - Reward: 0.61
State: 100111111111 - Reward: 1.40
State: 101000000000 - Reward: 0.47
State: 101000000001 - Reward: 1.31
State: 101000000010 - Reward: 1.41
State: 101000000011 - Reward: 0.00
State: 101000000100 - Reward: 0.95
State: 101000000101 - Reward: 0.27
State: 101000000110 - Reward: 0.45
State: 101000000111 - Reward: 1.36
State: 101000001000 - Reward: 0.02
State: 101000001001 - Reward: 1.39
State: 101000001010 - Reward: 1.63
State: 101000001011 - Reward: 1.98
```

```
State: 101000001100 - Reward: 0.84
State: 101000001101 - Reward: 0.26
State: 101000001110 - Reward: 0.14
State: 101000001111 - Reward: 0.77
State: 101000010000 - Reward: 1.46
State: 101000010001 - Reward: 0.20
State: 101000010010 - Reward: 0.63
State: 101000010011 - Reward: 1.76
State: 101000010100 - Reward: 0.27
State: 101000010101 - Reward: 1.55
State: 101000010110 - Reward: 1.51
State: 101000010111 - Reward: 0.27
State: 101000011000 - Reward: 1.99
State: 101000011001 - Reward: 0.29
State: 101000011010 - Reward: 1.06
State: 101000011011 - Reward: 0.02
State: 101000011100 - Reward: 1.30
State: 101000011101 - Reward: 0.88
State: 101000011110 - Reward: 1.44
State: 101000011111 - Reward: 1.26
State: 101000100000 - Reward: 0.30
State: 101000100001 - Reward: 0.82
State: 101000100010 - Reward: 1.37
State: 101000100011 - Reward: 1.72
State: 101000100100 - Reward: 0.17
State: 101000100101 - Reward: 0.20
State: 101000100110 - Reward: 1.50
State: 101000100111 - Reward: 1.18
State: 101000101000 - Reward: 0.77
State: 101000101001 - Reward: 1.93
State: 101000101010 - Reward: 0.63
State: 101000101011 - Reward: 0.28
State: 101000101100 - Reward: 0.55
State: 101000101101 - Reward: 0.17
State: 101000101110 - Reward: 1.11
State: 101000101111 - Reward: 1.20
State: 101000110000 - Reward: 1.22
State: 101000110001 - Reward: 1.56
State: 101000110010 - Reward: 1.38
State: 101000110011 - Reward: 1.70
State: 101000110100 - Reward: 1.32
State: 101000110101 - Reward: 0.60
State: 101000110110 - Reward: 1.04
State: 101000110111 - Reward: 1.02
State: 101000111000 - Reward: 1.50
State: 101000111001 - Reward: 0.59
State: 101000111010 - Reward: 0.11
State: 101000111011 - Reward: 1.80
State: 101000111100 - Reward: 1.91
State: 101000111101 - Reward: 0.99
State: 101000111110 - Reward: 0.23
State: 101000111111 - Reward: 1.00
```

```
State: 101001000000 - Reward: 1.19
State: 101001000001 - Reward: 1.06
State: 101001000010 - Reward: 1.96
State: 101001000011 - Reward: 1.97
State: 101001000100 - Reward: 1.87
State: 101001000101 - Reward: 0.26
State: 101001000110 - Reward: 1.72
State: 101001000111 - Reward: 1.14
State: 101001001000 - Reward: 0.73
State: 101001001001 - Reward: 1.37
State: 101001001010 - Reward: 1.53
State: 101001001011 - Reward: 1.91
State: 101001001100 - Reward: 1.54
State: 101001001101 - Reward: 0.03
State: 101001001110 - Reward: 0.14
State: 101001001111 - Reward: 0.52
State: 101001010000 - Reward: 0.08
State: 101001010001 - Reward: 0.12
State: 101001010010 - Reward: 1.58
State: 101001010011 - Reward: 1.01
State: 101001010100 - Reward: 1.26
State: 101001010101 - Reward: 1.00
State: 101001010110 - Reward: 0.83
State: 101001010111 - Reward: 1.40
State: 101001011000 - Reward: 0.16
State: 101001011001 - Reward: 1.07
State: 101001011010 - Reward: 1.23
State: 101001011011 - Reward: 0.55
State: 101001011100 - Reward: 0.62
State: 101001011101 - Reward: 1.02
State: 101001011110 - Reward: 0.41
State: 101001011111 - Reward: 1.62
State: 101001100000 - Reward: 1.07
State: 101001100001 - Reward: 0.78
State: 101001100010 - Reward: 1.27
State: 101001100011 - Reward: 1.67
State: 101001100100 - Reward: 1.36
State: 101001100101 - Reward: 0.13
State: 101001100110 - Reward: 1.40
State: 101001100111 - Reward: 1.46
State: 101001101000 - Reward: 1.69
State: 101001101001 - Reward: 0.12
State: 101001101010 - Reward: 0.17
State: 101001101011 - Reward: 0.87
State: 101001101100 - Reward: 0.91
State: 101001101101 - Reward: 1.22
State: 101001101110 - Reward: 0.62
State: 101001101111 - Reward: 1.48
State: 101001110000 - Reward: 1.48
State: 101001110001 - Reward: 0.24
State: 101001110010 - Reward: 1.42
State: 101001110011 - Reward: 1.40
```

```
State: 101001110100 - Reward: 0.33
State: 101001110101 - Reward: 1.91
State: 101001110110 - Reward: 1.05
State: 101001110111 - Reward: 1.57
State: 101001111000 - Reward: 1.44
State: 101001111001 - Reward: 0.33
State: 101001111010 - Reward: 0.25
State: 101001111011 - Reward: 1.56
State: 101001111100 - Reward: 0.54
State: 101001111101 - Reward: 1.77
State: 101001111110 - Reward: 1.54
State: 101001111111 - Reward: 0.06
State: 101010000000 - Reward: 1.61
State: 101010000001 - Reward: 0.54
State: 101010000010 - Reward: 0.13
State: 101010000011 - Reward: 1.42
State: 101010000100 - Reward: 1.15
State: 101010000101 - Reward: 0.15
State: 101010000110 - Reward: 0.91
State: 101010000111 - Reward: 0.72
State: 101010001000 - Reward: 1.00
State: 101010001001 - Reward: 1.13
State: 101010001010 - Reward: 0.74
State: 101010001011 - Reward: 0.51
State: 101010001100 - Reward: 0.21
State: 101010001101 - Reward: 1.15
State: 101010001110 - Reward: 1.45
State: 101010001111 - Reward: 0.46
State: 101010010000 - Reward: 1.02
State: 101010010001 - Reward: 0.09
State: 101010010010 - Reward: 1.73
State: 101010010011 - Reward: 0.49
State: 101010010100 - Reward: 0.94
State: 101010010101 - Reward: 0.77
State: 101010010110 - Reward: 0.30
State: 101010010111 - Reward: 1.86
State: 101010011000 - Reward: 1.71
State: 101010011001 - Reward: 1.11
State: 101010011010 - Reward: 1.83
State: 101010011011 - Reward: 1.48
State: 101010011100 - Reward: 0.84
State: 101010011101 - Reward: 0.64
State: 101010011110 - Reward: 0.83
State: 101010011111 - Reward: 1.44
State: 101010100000 - Reward: 0.54
State: 101010100001 - Reward: 0.16
State: 101010100010 - Reward: 0.75
State: 101010100011 - Reward: 1.00
State: 101010100100 - Reward: 1.80
State: 101010100101 - Reward: 0.36
State: 101010100110 - Reward: 1.61
State: 101010100111 - Reward: 1.96
```

```
State: 101010101000 - Reward: 1.91
State: 101010101001 - Reward: 0.14
State: 101010101010 - Reward: 0.93
State: 101010101011 - Reward: 0.56
State: 101010101100 - Reward: 1.69
State: 101010101101 - Reward: 0.65
State: 101010101110 - Reward: 1.11
State: 101010101111 - Reward: 0.02
State: 101010110000 - Reward: 0.40
State: 101010110001 - Reward: 1.13
State: 101010110010 - Reward: 0.61
State: 101010110011 - Reward: 1.25
State: 101010110100 - Reward: 0.93
State: 101010110101 - Reward: 1.18
State: 101010110110 - Reward: 0.99
State: 101010110111 - Reward: 1.55
State: 101010111000 - Reward: 0.39
State: 101010111001 - Reward: 1.80
State: 101010111010 - Reward: 1.52
State: 101010111011 - Reward: 0.49
State: 101010111100 - Reward: 0.01
State: 101010111101 - Reward: 0.82
State: 101010111110 - Reward: 0.47
State: 101010111111 - Reward: 0.69
State: 101011000000 - Reward: 1.68
State: 101011000001 - Reward: 1.75
State: 101011000010 - Reward: 1.90
State: 101011000011 - Reward: 0.00
State: 101011000100 - Reward: 1.31
State: 101011000101 - Reward: 1.70
State: 101011000110 - Reward: 1.45
State: 101011000111 - Reward: 0.21
State: 101011001000 - Reward: 1.06
State: 101011001001 - Reward: 0.48
State: 101011001010 - Reward: 0.98
State: 101011001011 - Reward: 0.12
State: 101011001100 - Reward: 1.99
State: 101011001101 - Reward: 1.42
State: 101011001110 - Reward: 0.19
State: 101011001111 - Reward: 1.84
State: 101011010000 - Reward: 1.79
State: 101011010001 - Reward: 1.04
State: 101011010010 - Reward: 1.40
State: 101011010011 - Reward: 0.74
State: 101011010100 - Reward: 1.95
State: 101011010101 - Reward: 0.17
State: 101011010110 - Reward: 0.19
State: 101011010111 - Reward: 0.27
State: 101011011000 - Reward: 1.64
State: 101011011001 - Reward: 0.15
State: 101011011010 - Reward: 1.14
State: 101011011011 - Reward: 0.87
```

```
State: 101011011100 - Reward: 1.93
State: 101011011101 - Reward: 0.47
State: 101011011110 - Reward: 0.52
State: 101011011111 - Reward: 0.63
State: 101011100000 - Reward: 1.60
State: 101011100001 - Reward: 1.40
State: 101011100010 - Reward: 1.47
State: 101011100011 - Reward: 0.64
State: 101011100100 - Reward: 0.54
State: 101011100101 - Reward: 0.15
State: 101011100110 - Reward: 0.41
State: 101011100111 - Reward: 1.56
State: 101011101000 - Reward: 1.17
State: 101011101001 - Reward: 0.31
State: 101011101010 - Reward: 0.33
State: 101011101011 - Reward: 0.93
State: 101011101100 - Reward: 0.81
State: 101011101101 - Reward: 1.07
State: 101011101110 - Reward: 1.93
State: 101011101111 - Reward: 0.42
State: 101011110000 - Reward: 0.62
State: 101011110001 - Reward: 0.53
State: 101011110010 - Reward: 0.24
State: 101011110011 - Reward: 0.32
State: 101011110100 - Reward: 1.37
State: 101011110101 - Reward: 1.65
State: 101011110110 - Reward: 1.39
State: 101011110111 - Reward: 0.08
State: 101011111000 - Reward: 1.67
State: 101011111001 - Reward: 0.66
State: 101011111010 - Reward: 0.18
State: 101011111011 - Reward: 0.50
State: 101011111100 - Reward: 0.71
State: 101011111101 - Reward: 1.03
State: 101011111110 - Reward: 1.35
State: 101011111111 - Reward: 0.52
State: 101100000000 - Reward: 1.98
State: 101100000001 - Reward: 0.06
State: 101100000010 - Reward: 0.81
State: 101100000011 - Reward: 0.90
State: 101100000100 - Reward: 1.50
State: 101100000101 - Reward: 0.50
State: 101100000110 - Reward: 0.92
State: 101100000111 - Reward: 1.61
State: 101100001000 - Reward: 0.28
State: 101100001001 - Reward: 0.02
State: 101100001010 - Reward: 1.66
State: 101100001011 - Reward: 1.97
State: 101100001100 - Reward: 0.26
State: 101100001101 - Reward: 1.65
State: 101100001110 - Reward: 0.74
State: 101100001111 - Reward: 1.26
```

```
State: 101100010000 - Reward: 1.29
State: 101100010001 - Reward: 1.16
State: 101100010010 - Reward: 0.52
State: 101100010011 - Reward: 1.63
State: 101100010100 - Reward: 0.04
State: 101100010101 - Reward: 0.13
State: 101100010110 - Reward: 1.80
State: 101100010111 - Reward: 0.89
State: 101100011000 - Reward: 0.26
State: 101100011001 - Reward: 1.81
State: 101100011010 - Reward: 1.66
State: 101100011011 - Reward: 0.66
State: 101100011100 - Reward: 0.09
State: 101100011101 - Reward: 0.92
State: 101100011110 - Reward: 0.34
State: 101100011111 - Reward: 1.15
State: 101100100000 - Reward: 1.64
State: 101100100001 - Reward: 0.79
State: 101100100010 - Reward: 0.06
State: 101100100011 - Reward: 1.37
State: 101100100100 - Reward: 0.35
State: 101100100101 - Reward: 0.43
State: 101100100110 - Reward: 0.37
State: 101100100111 - Reward: 0.56
State: 101100101000 - Reward: 1.77
State: 101100101001 - Reward: 0.07
State: 101100101010 - Reward: 1.24
State: 101100101011 - Reward: 0.49
State: 101100101100 - Reward: 0.59
State: 101100101101 - Reward: 0.82
State: 101100101110 - Reward: 1.10
State: 101100101111 - Reward: 0.12
State: 101100110000 - Reward: 0.56
State: 101100110001 - Reward: 0.27
State: 101100110010 - Reward: 0.40
State: 101100110011 - Reward: 1.77
State: 101100110100 - Reward: 1.05
State: 101100110101 - Reward: 1.26
State: 101100110110 - Reward: 1.60
State: 101100110111 - Reward: 1.59
State: 101100111000 - Reward: 1.98
State: 101100111001 - Reward: 1.56
State: 101100111010 - Reward: 0.72
State: 101100111011 - Reward: 1.09
State: 101100111100 - Reward: 0.97
State: 101100111101 - Reward: 1.83
State: 101100111110 - Reward: 1.00
State: 101100111111 - Reward: 0.78
State: 101101000000 - Reward: 0.36
State: 101101000001 - Reward: 0.64
State: 101101000010 - Reward: 0.44
State: 101101000011 - Reward: 1.79
```

```
State: 101101000100 - Reward: 1.56
State: 101101000101 - Reward: 0.12
State: 101101000110 - Reward: 1.98
State: 101101000111 - Reward: 1.06
State: 101101001000 - Reward: 1.53
State: 101101001001 - Reward: 2.00
State: 101101001010 - Reward: 1.95
State: 101101001011 - Reward: 0.20
State: 101101001100 - Reward: 1.31
State: 101101001101 - Reward: 0.53
State: 101101001110 - Reward: 1.63
State: 101101001111 - Reward: 1.83
State: 101101010000 - Reward: 0.11
State: 101101010001 - Reward: 1.99
State: 101101010010 - Reward: 0.44
State: 101101010011 - Reward: 1.69
State: 101101010100 - Reward: 1.59
State: 101101010101 - Reward: 0.71
State: 101101010110 - Reward: 1.68
State: 101101010111 - Reward: 1.69
State: 101101011000 - Reward: 0.35
State: 101101011001 - Reward: 1.19
State: 101101011010 - Reward: 1.61
State: 101101011011 - Reward: 1.40
State: 101101011100 - Reward: 1.83
State: 101101011101 - Reward: 0.06
State: 101101011110 - Reward: 1.40
State: 101101011111 - Reward: 1.90
State: 101101100000 - Reward: 1.13
State: 101101100001 - Reward: 1.13
State: 101101100010 - Reward: 0.38
State: 101101100011 - Reward: 1.98
State: 101101100100 - Reward: 1.76
State: 101101100101 - Reward: 0.98
State: 101101100110 - Reward: 0.62
State: 101101100111 - Reward: 0.98
State: 101101101000 - Reward: 0.18
State: 101101101001 - Reward: 0.47
State: 101101101010 - Reward: 0.44
State: 101101101011 - Reward: 1.05
State: 101101101100 - Reward: 0.00
State: 101101101101 - Reward: 1.84
State: 101101101110 - Reward: 0.40
State: 101101101111 - Reward: 0.26
State: 101101110000 - Reward: 1.43
State: 101101110001 - Reward: 1.84
State: 101101110010 - Reward: 1.69
State: 101101110011 - Reward: 0.65
State: 101101110100 - Reward: 0.04
State: 101101110101 - Reward: 1.17
State: 101101110110 - Reward: 1.83
State: 101101110111 - Reward: 1.55
```

```
State: 101101111000 - Reward: 1.69
State: 101101111001 - Reward: 1.72
State: 101101111010 - Reward: 1.92
State: 101101111011 - Reward: 0.75
State: 101101111100 - Reward: 1.88
State: 101101111101 - Reward: 0.79
State: 101101111110 - Reward: 0.20
State: 101101111111 - Reward: 0.60
State: 101110000000 - Reward: 0.27
State: 101110000001 - Reward: 0.32
State: 101110000010 - Reward: 1.90
State: 101110000011 - Reward: 1.58
State: 101110000100 - Reward: 1.92
State: 101110000101 - Reward: 1.30
State: 101110000110 - Reward: 0.35
State: 101110000111 - Reward: 1.94
State: 101110001000 - Reward: 1.39
State: 101110001001 - Reward: 1.86
State: 101110001010 - Reward: 1.57
State: 101110001011 - Reward: 0.45
State: 101110001100 - Reward: 1.18
State: 101110001101 - Reward: 0.35
State: 101110001110 - Reward: 0.61
State: 101110001111 - Reward: 1.38
State: 101110010000 - Reward: 0.25
State: 101110010001 - Reward: 1.46
State: 101110010010 - Reward: 1.90
State: 101110010011 - Reward: 1.90
State: 101110010100 - Reward: 0.78
State: 101110010101 - Reward: 1.99
State: 101110010110 - Reward: 1.93
State: 101110010111 - Reward: 0.06
State: 101110011000 - Reward: 1.20
State: 101110011001 - Reward: 1.84
State: 101110011010 - Reward: 1.94
State: 101110011011 - Reward: 0.44
State: 101110011100 - Reward: 1.13
State: 101110011101 - Reward: 1.87
State: 101110011110 - Reward: 0.28
State: 101110011111 - Reward: 1.49
State: 101110100000 - Reward: 0.48
State: 101110100001 - Reward: 1.96
State: 101110100010 - Reward: 0.34
State: 101110100011 - Reward: 1.77
State: 101110100100 - Reward: 0.18
State: 101110100101 - Reward: 1.42
State: 101110100110 - Reward: 1.28
State: 101110100111 - Reward: 1.77
State: 101110101000 - Reward: 0.89
State: 101110101001 - Reward: 0.53
State: 101110101010 - Reward: 0.50
State: 101110101011 - Reward: 0.14
```

```
State: 101110101100 - Reward: 0.51
State: 101110101101 - Reward: 0.22
State: 101110101110 - Reward: 0.00
State: 101110101111 - Reward: 0.77
State: 101110110000 - Reward: 1.47
State: 101110110001 - Reward: 1.94
State: 101110110010 - Reward: 1.77
State: 101110110011 - Reward: 0.99
State: 101110110100 - Reward: 0.76
State: 101110110101 - Reward: 1.09
State: 101110110110 - Reward: 0.20
State: 101110110111 - Reward: 0.96
State: 101110111000 - Reward: 1.73
State: 101110111001 - Reward: 1.30
State: 101110111010 - Reward: 1.37
State: 101110111011 - Reward: 0.33
State: 101110111100 - Reward: 0.15
State: 101110111101 - Reward: 1.69
State: 101110111110 - Reward: 0.59
State: 101110111111 - Reward: 0.64
State: 101111000000 - Reward: 1.90
State: 101111000001 - Reward: 0.15
State: 101111000010 - Reward: 0.34
State: 101111000011 - Reward: 0.75
State: 101111000100 - Reward: 1.46
State: 101111000101 - Reward: 1.09
State: 101111000110 - Reward: 1.80
State: 101111000111 - Reward: 0.19
State: 101111001000 - Reward: 1.19
State: 101111001001 - Reward: 1.23
State: 101111001010 - Reward: 0.97
State: 101111001011 - Reward: 0.06
State: 101111001100 - Reward: 1.88
State: 101111001101 - Reward: 0.33
State: 101111001110 - Reward: 1.78
State: 101111001111 - Reward: 0.31
State: 101111010000 - Reward: 0.20
State: 101111010001 - Reward: 0.41
State: 101111010010 - Reward: 0.38
State: 101111010011 - Reward: 1.39
State: 101111010100 - Reward: 1.44
State: 101111010101 - Reward: 1.46
State: 101111010110 - Reward: 0.53
State: 101111010111 - Reward: 0.56
State: 101111011000 - Reward: 0.48
State: 101111011001 - Reward: 0.10
State: 101111011010 - Reward: 1.14
State: 101111011011 - Reward: 1.68
State: 101111011100 - Reward: 0.31
State: 101111011101 - Reward: 0.72
State: 101111011110 - Reward: 0.86
State: 101111011111 - Reward: 0.59
```

```
State: 101111100000 - Reward: 1.32
State: 101111100001 - Reward: 1.20
State: 101111100010 - Reward: 0.40
State: 101111100011 - Reward: 0.05
State: 101111100100 - Reward: 0.34
State: 101111100101 - Reward: 0.58
State: 101111100110 - Reward: 0.16
State: 101111100111 - Reward: 1.69
State: 101111101000 - Reward: 0.62
State: 101111101001 - Reward: 0.79
State: 101111101010 - Reward: 0.98
State: 101111101011 - Reward: 1.32
State: 101111101100 - Reward: 0.18
State: 101111101101 - Reward: 1.09
State: 101111101110 - Reward: 0.37
State: 101111101111 - Reward: 1.77
State: 101111110000 - Reward: 0.74
State: 101111110001 - Reward: 0.89
State: 101111110010 - Reward: 0.53
State: 101111110011 - Reward: 0.93
State: 101111110100 - Reward: 0.45
State: 101111110101 - Reward: 0.54
State: 101111110110 - Reward: 0.12
State: 101111110111 - Reward: 1.50
State: 101111111000 - Reward: 1.33
State: 101111111001 - Reward: 0.17
State: 101111111010 - Reward: 0.69
State: 101111111011 - Reward: 1.08
State: 101111111100 - Reward: 1.94
State: 101111111101 - Reward: 1.18
State: 101111111110 - Reward: 1.11
State: 101111111111 - Reward: 1.68
State: 110000000000 - Reward: 1.64
State: 110000000001 - Reward: 0.84
State: 110000000010 - Reward: 1.07
State: 110000000011 - Reward: 1.73
State: 110000000100 - Reward: 0.95
State: 110000000101 - Reward: 1.76
State: 110000000110 - Reward: 0.95
State: 110000000111 - Reward: 0.16
State: 110000001000 - Reward: 1.81
State: 110000001001 - Reward: 1.43
State: 110000001010 - Reward: 1.00
State: 110000001011 - Reward: 1.80
State: 110000001100 - Reward: 1.60
State: 110000001101 - Reward: 1.36
State: 110000001110 - Reward: 1.24
State: 110000001111 - Reward: 0.24
State: 110000010000 - Reward: 1.51
State: 110000010001 - Reward: 0.35
State: 110000010010 - Reward: 1.97
State: 110000010011 - Reward: 1.94
```

```
State: 110000010100 - Reward: 1.62
State: 110000010101 - Reward: 0.25
State: 110000010110 - Reward: 0.85
State: 110000010111 - Reward: 1.98
State: 110000011000 - Reward: 0.87
State: 110000011001 - Reward: 1.99
State: 110000011010 - Reward: 1.25
State: 110000011011 - Reward: 1.67
State: 110000011100 - Reward: 0.52
State: 110000011101 - Reward: 1.82
State: 110000011110 - Reward: 1.83
State: 110000011111 - Reward: 0.13
State: 110000100000 - Reward: 0.78
State: 110000100001 - Reward: 0.79
State: 110000100010 - Reward: 0.65
State: 110000100011 - Reward: 0.55
State: 110000100100 - Reward: 0.92
State: 110000100101 - Reward: 1.75
State: 110000100110 - Reward: 1.57
State: 110000100111 - Reward: 1.25
State: 110000101000 - Reward: 1.05
State: 110000101001 - Reward: 0.84
State: 110000101010 - Reward: 0.83
State: 110000101011 - Reward: 0.30
State: 110000101100 - Reward: 1.18
State: 110000101101 - Reward: 1.52
State: 110000101110 - Reward: 1.88
State: 110000101111 - Reward: 1.85
State: 110000110000 - Reward: 1.13
State: 110000110001 - Reward: 0.20
State: 110000110010 - Reward: 0.57
State: 110000110011 - Reward: 1.07
State: 110000110100 - Reward: 0.69
State: 110000110101 - Reward: 0.82
State: 110000110110 - Reward: 0.77
State: 110000110111 - Reward: 0.97
State: 110000111000 - Reward: 1.22
State: 110000111001 - Reward: 0.07
State: 110000111010 - Reward: 0.55
State: 110000111011 - Reward: 0.29
State: 110000111100 - Reward: 1.22
State: 110000111101 - Reward: 1.39
State: 110000111110 - Reward: 0.08
State: 110000111111 - Reward: 1.78
State: 110001000000 - Reward: 0.66
State: 110001000001 - Reward: 0.48
State: 110001000010 - Reward: 1.49
State: 110001000011 - Reward: 1.84
State: 110001000100 - Reward: 1.79
State: 110001000101 - Reward: 0.04
State: 110001000110 - Reward: 1.63
State: 110001000111 - Reward: 0.61
```

```
State: 110001001000 - Reward: 0.56
State: 110001001001 - Reward: 0.98
State: 110001001010 - Reward: 1.39
State: 110001001011 - Reward: 0.20
State: 110001001100 - Reward: 1.74
State: 110001001101 - Reward: 0.27
State: 110001001110 - Reward: 1.95
State: 110001001111 - Reward: 0.89
State: 110001010000 - Reward: 1.65
State: 110001010001 - Reward: 0.54
State: 110001010010 - Reward: 0.83
State: 110001010011 - Reward: 1.29
State: 110001010100 - Reward: 0.38
State: 110001010101 - Reward: 0.42
State: 110001010110 - Reward: 1.65
State: 110001010111 - Reward: 1.48
State: 110001011000 - Reward: 1.52
State: 110001011001 - Reward: 1.73
State: 110001011010 - Reward: 1.64
State: 110001011011 - Reward: 1.03
State: 110001011100 - Reward: 0.32
State: 110001011101 - Reward: 0.62
State: 110001011110 - Reward: 1.01
State: 110001011111 - Reward: 0.27
State: 110001100000 - Reward: 1.70
State: 110001100001 - Reward: 1.76
State: 110001100010 - Reward: 0.06
State: 110001100011 - Reward: 0.39
State: 110001100100 - Reward: 1.67
State: 110001100101 - Reward: 1.67
State: 110001100110 - Reward: 0.50
State: 110001100111 - Reward: 0.91
State: 110001101000 - Reward: 1.84
State: 110001101001 - Reward: 1.41
State: 110001101010 - Reward: 0.55
State: 110001101011 - Reward: 1.65
State: 110001101100 - Reward: 1.01
State: 110001101101 - Reward: 1.27
State: 110001101110 - Reward: 0.25
State: 110001101111 - Reward: 0.06
State: 110001110000 - Reward: 0.74
State: 110001110001 - Reward: 1.19
State: 110001110010 - Reward: 0.36
State: 110001110011 - Reward: 1.74
State: 110001110100 - Reward: 1.17
State: 110001110101 - Reward: 0.70
State: 110001110110 - Reward: 0.33
State: 110001110111 - Reward: 1.79
State: 110001111000 - Reward: 1.50
State: 110001111001 - Reward: 1.38
State: 110001111010 - Reward: 0.57
State: 110001111011 - Reward: 0.77
```

```
State: 110001111100 - Reward: 0.33
State: 110001111101 - Reward: 1.14
State: 110001111110 - Reward: 1.93
State: 110001111111 - Reward: 1.71
State: 110010000000 - Reward: 1.29
State: 110010000001 - Reward: 1.36
State: 110010000010 - Reward: 0.54
State: 110010000011 - Reward: 0.82
State: 110010000100 - Reward: 0.04
State: 110010000101 - Reward: 1.56
State: 110010000110 - Reward: 1.54
State: 110010000111 - Reward: 0.02
State: 110010001000 - Reward: 1.82
State: 110010001001 - Reward: 1.29
State: 110010001010 - Reward: 1.20
State: 110010001011 - Reward: 0.02
State: 110010001100 - Reward: 0.50
State: 110010001101 - Reward: 1.61
State: 110010001110 - Reward: 0.61
State: 110010001111 - Reward: 1.93
State: 110010010000 - Reward: 1.29
State: 110010010001 - Reward: 0.85
State: 110010010010 - Reward: 0.75
State: 110010010011 - Reward: 0.70
State: 110010010100 - Reward: 0.50
State: 110010010101 - Reward: 0.93
State: 110010010110 - Reward: 1.35
State: 110010010111 - Reward: 1.65
State: 110010011000 - Reward: 0.79
State: 110010011001 - Reward: 0.20
State: 110010011010 - Reward: 1.02
State: 110010011011 - Reward: 1.32
State: 110010011100 - Reward: 1.69
State: 110010011101 - Reward: 0.75
State: 110010011110 - Reward: 1.29
State: 110010011111 - Reward: 1.22
State: 110010100000 - Reward: 0.60
State: 110010100001 - Reward: 0.22
State: 110010100010 - Reward: 0.13
State: 110010100011 - Reward: 1.98
State: 110010100100 - Reward: 1.28
State: 110010100101 - Reward: 1.72
State: 110010100110 - Reward: 0.52
State: 110010100111 - Reward: 1.42
State: 110010101000 - Reward: 1.78
State: 110010101001 - Reward: 0.60
State: 110010101010 - Reward: 0.30
State: 110010101011 - Reward: 1.53
State: 110010101100 - Reward: 1.80
State: 110010101101 - Reward: 1.61
State: 110010101110 - Reward: 1.60
State: 110010101111 - Reward: 1.20
```

```
State: 110010110000 - Reward: 1.32
State: 110010110001 - Reward: 1.36
State: 110010110010 - Reward: 1.44
State: 110010110011 - Reward: 1.31
State: 110010110100 - Reward: 1.99
State: 110010110101 - Reward: 0.52
State: 110010110110 - Reward: 0.84
State: 110010110111 - Reward: 0.78
State: 110010111000 - Reward: 0.07
State: 110010111001 - Reward: 1.42
State: 110010111010 - Reward: 1.14
State: 110010111011 - Reward: 0.38
State: 110010111100 - Reward: 1.45
State: 110010111101 - Reward: 0.44
State: 110010111110 - Reward: 1.07
State: 110010111111 - Reward: 1.57
State: 110011000000 - Reward: 1.81
State: 110011000001 - Reward: 1.34
State: 110011000010 - Reward: 1.01
State: 110011000011 - Reward: 1.69
State: 110011000100 - Reward: 1.68
State: 110011000101 - Reward: 1.75
State: 110011000110 - Reward: 0.36
State: 110011000111 - Reward: 0.20
State: 110011001000 - Reward: 0.26
State: 110011001001 - Reward: 0.52
State: 110011001010 - Reward: 1.62
State: 110011001011 - Reward: 1.53
State: 110011001100 - Reward: 0.37
State: 110011001101 - Reward: 1.36
State: 110011001110 - Reward: 0.67
State: 110011001111 - Reward: 0.18
State: 110011010000 - Reward: 0.71
State: 110011010001 - Reward: 1.49
State: 110011010010 - Reward: 0.61
State: 110011010011 - Reward: 1.58
State: 110011010100 - Reward: 0.66
State: 110011010101 - Reward: 0.52
State: 110011010110 - Reward: 0.59
State: 110011010111 - Reward: 1.70
State: 110011011000 - Reward: 0.94
State: 110011011001 - Reward: 1.73
State: 110011011010 - Reward: 1.17
State: 110011011011 - Reward: 1.89
State: 110011011100 - Reward: 0.14
State: 110011011101 - Reward: 1.78
State: 110011011110 - Reward: 1.00
State: 110011011111 - Reward: 1.73
State: 110011100000 - Reward: 0.76
State: 110011100001 - Reward: 0.60
State: 110011100010 - Reward: 0.11
State: 110011100011 - Reward: 1.71
```

```
State: 110011100100 - Reward: 0.27
State: 110011100101 - Reward: 0.40
State: 110011100110 - Reward: 0.82
State: 110011100111 - Reward: 1.14
State: 110011101000 - Reward: 1.81
State: 110011101001 - Reward: 0.92
State: 110011101010 - Reward: 0.63
State: 110011101011 - Reward: 1.43
State: 110011101100 - Reward: 1.56
State: 110011101101 - Reward: 0.98
State: 110011101110 - Reward: 1.26
State: 110011101111 - Reward: 0.35
State: 110011110000 - Reward: 1.27
State: 110011110001 - Reward: 0.01
State: 110011110010 - Reward: 0.55
State: 110011110011 - Reward: 1.52
State: 110011110100 - Reward: 0.34
State: 110011110101 - Reward: 1.53
State: 110011110110 - Reward: 0.98
State: 110011110111 - Reward: 1.53
State: 110011111000 - Reward: 0.18
State: 110011111001 - Reward: 1.23
State: 110011111010 - Reward: 1.27
State: 110011111011 - Reward: 0.81
State: 110011111100 - Reward: 1.93
State: 110011111101 - Reward: 0.77
State: 110011111110 - Reward: 0.08
State: 110011111111 - Reward: 0.40
State: 110100000000 - Reward: 0.75
State: 110100000001 - Reward: 0.03
State: 110100000010 - Reward: 0.64
State: 110100000011 - Reward: 1.67
State: 110100000100 - Reward: 0.38
State: 110100000101 - Reward: 1.35
State: 110100000110 - Reward: 1.25
State: 110100000111 - Reward: 0.50
State: 110100001000 - Reward: 1.39
State: 110100001001 - Reward: 0.69
State: 110100001010 - Reward: 0.26
State: 110100001011 - Reward: 0.77
State: 110100001100 - Reward: 1.18
State: 110100001101 - Reward: 0.33
State: 110100001110 - Reward: 1.65
State: 110100001111 - Reward: 0.60
State: 110100010000 - Reward: 0.58
State: 110100010001 - Reward: 1.46
State: 110100010010 - Reward: 1.19
State: 110100010011 - Reward: 0.68
State: 110100010100 - Reward: 1.78
State: 110100010101 - Reward: 1.99
State: 110100010110 - Reward: 0.69
State: 110100010111 - Reward: 1.80
```

```
State: 110100011000 - Reward: 0.72
State: 110100011001 - Reward: 0.38
State: 110100011010 - Reward: 1.90
State: 110100011011 - Reward: 1.84
State: 110100011100 - Reward: 0.81
State: 110100011101 - Reward: 0.46
State: 110100011110 - Reward: 1.45
State: 110100011111 - Reward: 0.26
State: 110100100000 - Reward: 1.47
State: 110100100001 - Reward: 1.18
State: 110100100010 - Reward: 0.34
State: 110100100011 - Reward: 0.73
State: 110100100100 - Reward: 1.30
State: 110100100101 - Reward: 0.07
State: 110100100110 - Reward: 1.75
State: 110100100111 - Reward: 0.51
State: 110100101000 - Reward: 1.07
State: 110100101001 - Reward: 0.10
State: 110100101010 - Reward: 1.99
State: 110100101011 - Reward: 1.32
State: 110100101100 - Reward: 1.31
State: 110100101101 - Reward: 0.04
State: 110100101110 - Reward: 1.38
State: 110100101111 - Reward: 0.83
State: 110100110000 - Reward: 0.76
State: 110100110001 - Reward: 1.09
State: 110100110010 - Reward: 0.95
State: 110100110011 - Reward: 0.31
State: 110100110100 - Reward: 1.39
State: 110100110101 - Reward: 1.26
State: 110100110110 - Reward: 0.60
State: 110100110111 - Reward: 1.32
State: 110100111000 - Reward: 1.32
State: 110100111001 - Reward: 0.54
State: 110100111010 - Reward: 1.21
State: 110100111011 - Reward: 0.27
State: 110100111100 - Reward: 1.66
State: 110100111101 - Reward: 0.21
State: 110100111110 - Reward: 1.44
State: 110100111111 - Reward: 0.24
State: 110101000000 - Reward: 0.23
State: 110101000001 - Reward: 0.21
State: 110101000010 - Reward: 0.40
State: 110101000011 - Reward: 0.40
State: 110101000100 - Reward: 0.53
State: 110101000101 - Reward: 1.05
State: 110101000110 - Reward: 0.40
State: 110101000111 - Reward: 1.41
State: 110101001000 - Reward: 0.59
State: 110101001001 - Reward: 0.08
State: 110101001010 - Reward: 0.99
State: 110101001011 - Reward: 0.42
```

```
State: 110101001100 - Reward: 1.87
State: 110101001101 - Reward: 0.66
State: 110101001110 - Reward: 0.01
State: 110101001111 - Reward: 1.34
State: 110101010000 - Reward: 1.81
State: 110101010001 - Reward: 1.67
State: 110101010010 - Reward: 1.34
State: 110101010011 - Reward: 0.30
State: 110101010100 - Reward: 0.18
State: 110101010101 - Reward: 1.02
State: 110101010110 - Reward: 1.45
State: 110101010111 - Reward: 0.20
State: 110101011000 - Reward: 0.51
State: 110101011001 - Reward: 0.46
State: 110101011010 - Reward: 1.98
State: 110101011011 - Reward: 0.59
State: 110101011100 - Reward: 0.93
State: 110101011101 - Reward: 0.20
State: 110101011110 - Reward: 0.35
State: 110101011111 - Reward: 0.08
State: 110101100000 - Reward: 0.58
State: 110101100001 - Reward: 1.60
State: 110101100010 - Reward: 0.63
State: 110101100011 - Reward: 1.48
State: 110101100100 - Reward: 0.19
State: 110101100101 - Reward: 1.52
State: 110101100110 - Reward: 0.09
State: 110101100111 - Reward: 1.70
State: 110101101000 - Reward: 1.33
State: 110101101001 - Reward: 0.34
State: 110101101010 - Reward: 0.72
State: 110101101011 - Reward: 0.88
State: 110101101100 - Reward: 1.24
State: 110101101101 - Reward: 1.76
State: 110101101110 - Reward: 0.19
State: 110101101111 - Reward: 1.63
State: 110101110000 - Reward: 0.37
State: 110101110001 - Reward: 0.80
State: 110101110010 - Reward: 1.92
State: 110101110011 - Reward: 0.54
State: 110101110100 - Reward: 0.77
State: 110101110101 - Reward: 1.70
State: 110101110110 - Reward: 1.60
State: 110101110111 - Reward: 1.30
State: 110101111000 - Reward: 1.59
State: 110101111001 - Reward: 0.23
State: 110101111010 - Reward: 1.39
State: 110101111011 - Reward: 0.12
State: 110101111100 - Reward: 1.88
State: 110101111101 - Reward: 0.32
State: 110101111110 - Reward: 0.83
State: 110101111111 - Reward: 1.18
```

```
State: 110110000000 - Reward: 1.60
State: 110110000001 - Reward: 1.36
State: 110110000010 - Reward: 0.36
State: 110110000011 - Reward: 0.76
State: 110110000100 - Reward: 0.72
State: 110110000101 - Reward: 0.06
State: 110110000110 - Reward: 1.37
State: 110110000111 - Reward: 1.68
State: 110110001000 - Reward: 1.95
State: 110110001001 - Reward: 0.26
State: 110110001010 - Reward: 1.84
State: 110110001011 - Reward: 0.23
State: 110110001100 - Reward: 0.82
State: 110110001101 - Reward: 0.09
State: 110110001110 - Reward: 0.52
State: 110110001111 - Reward: 0.63
State: 110110010000 - Reward: 1.41
State: 110110010001 - Reward: 1.36
State: 110110010010 - Reward: 1.54
State: 110110010011 - Reward: 1.15
State: 110110010100 - Reward: 1.13
State: 110110010101 - Reward: 1.96
State: 110110010110 - Reward: 1.34
State: 110110010111 - Reward: 0.68
State: 110110011000 - Reward: 1.05
State: 110110011001 - Reward: 1.40
State: 110110011010 - Reward: 0.19
State: 110110011011 - Reward: 1.32
State: 110110011100 - Reward: 0.50
State: 110110011101 - Reward: 0.69
State: 110110011110 - Reward: 1.35
State: 110110011111 - Reward: 0.77
State: 110110100000 - Reward: 1.68
State: 110110100001 - Reward: 1.12
State: 110110100010 - Reward: 1.98
State: 110110100011 - Reward: 0.11
State: 110110100100 - Reward: 1.29
State: 110110100101 - Reward: 0.31
State: 110110100110 - Reward: 1.70
State: 110110100111 - Reward: 1.70
State: 110110101000 - Reward: 1.74
State: 110110101001 - Reward: 0.15
State: 110110101010 - Reward: 0.98
State: 110110101011 - Reward: 0.48
State: 110110101100 - Reward: 1.94
State: 110110101101 - Reward: 0.10
State: 110110101110 - Reward: 0.45
State: 110110101111 - Reward: 1.29
State: 110110110000 - Reward: 0.81
State: 110110110001 - Reward: 0.47
State: 110110110010 - Reward: 0.92
State: 110110110011 - Reward: 1.60
```

```
State: 110110110100 - Reward: 0.90
State: 110110110101 - Reward: 1.71
State: 110110110110 - Reward: 0.89
State: 110110110111 - Reward: 0.24
State: 110110111000 - Reward: 0.99
State: 110110111001 - Reward: 1.31
State: 110110111010 - Reward: 0.21
State: 110110111011 - Reward: 0.82
State: 110110111100 - Reward: 1.11
State: 110110111101 - Reward: 0.00
State: 110110111110 - Reward: 0.18
State: 110110111111 - Reward: 1.21
State: 110111000000 - Reward: 1.24
State: 110111000001 - Reward: 0.61
State: 110111000010 - Reward: 1.02
State: 110111000011 - Reward: 0.41
State: 110111000100 - Reward: 1.34
State: 110111000101 - Reward: 1.90
State: 110111000110 - Reward: 0.73
State: 110111000111 - Reward: 0.11
State: 110111001000 - Reward: 0.45
State: 110111001001 - Reward: 0.91
State: 110111001010 - Reward: 1.12
State: 110111001011 - Reward: 1.24
State: 110111001100 - Reward: 0.95
State: 110111001101 - Reward: 1.31
State: 110111001110 - Reward: 1.43
State: 110111001111 - Reward: 0.23
State: 110111010000 - Reward: 1.52
State: 110111010001 - Reward: 0.44
State: 110111010010 - Reward: 0.68
State: 110111010011 - Reward: 1.66
State: 110111010100 - Reward: 1.93
State: 110111010101 - Reward: 0.58
State: 110111010110 - Reward: 1.04
State: 110111010111 - Reward: 1.40
State: 110111011000 - Reward: 0.09
State: 110111011001 - Reward: 0.33
State: 110111011010 - Reward: 0.28
State: 110111011011 - Reward: 1.43
State: 110111011100 - Reward: 1.44
State: 110111011101 - Reward: 0.21
State: 110111011110 - Reward: 1.22
State: 110111011111 - Reward: 0.38
State: 110111100000 - Reward: 1.86
State: 110111100001 - Reward: 0.79
State: 110111100010 - Reward: 0.91
State: 110111100011 - Reward: 1.56
State: 110111100100 - Reward: 1.43
State: 110111100101 - Reward: 0.22
State: 110111100110 - Reward: 0.83
State: 110111100111 - Reward: 1.85
```

```
State: 110111101000 - Reward: 1.67
State: 110111101001 - Reward: 1.18
State: 110111101010 - Reward: 1.54
State: 110111101011 - Reward: 0.90
State: 110111101100 - Reward: 1.32
State: 110111101101 - Reward: 1.91
State: 110111101110 - Reward: 0.27
State: 110111101111 - Reward: 1.00
State: 110111110000 - Reward: 1.06
State: 110111110001 - Reward: 0.10
State: 110111110010 - Reward: 1.87
State: 110111110011 - Reward: 1.68
State: 110111110100 - Reward: 0.97
State: 110111110101 - Reward: 1.02
State: 110111110110 - Reward: 1.84
State: 110111110111 - Reward: 0.35
State: 110111111000 - Reward: 1.16
State: 110111111001 - Reward: 1.46
State: 110111111010 - Reward: 0.26
State: 110111111011 - Reward: 0.77
State: 110111111100 - Reward: 1.20
State: 110111111101 - Reward: 1.77
State: 110111111110 - Reward: 1.01
State: 110111111111 - Reward: 0.77
State: 111000000000 - Reward: 1.96
State: 111000000001 - Reward: 1.83
State: 111000000010 - Reward: 1.52
State: 111000000011 - Reward: 0.55
State: 111000000100 - Reward: 1.93
State: 111000000101 - Reward: 1.94
State: 111000000110 - Reward: 0.91
State: 111000000111 - Reward: 0.27
State: 111000001000 - Reward: 0.83
State: 111000001001 - Reward: 1.40
State: 111000001010 - Reward: 1.50
State: 111000001011 - Reward: 0.60
State: 111000001100 - Reward: 1.40
State: 111000001101 - Reward: 1.72
State: 111000001110 - Reward: 1.42
State: 111000001111 - Reward: 1.87
State: 111000010000 - Reward: 1.27
State: 111000010001 - Reward: 0.40
State: 111000010010 - Reward: 1.25
State: 111000010011 - Reward: 0.58
State: 111000010100 - Reward: 0.69
State: 111000010101 - Reward: 1.34
State: 111000010110 - Reward: 1.96
State: 111000010111 - Reward: 1.30
State: 111000011000 - Reward: 1.92
State: 111000011001 - Reward: 1.01
State: 111000011010 - Reward: 1.39
State: 111000011011 - Reward: 0.64
```

```
State: 111000011100 - Reward: 0.23
State: 111000011101 - Reward: 0.70
State: 111000011110 - Reward: 0.96
State: 111000011111 - Reward: 1.14
State: 111000100000 - Reward: 1.33
State: 111000100001 - Reward: 0.83
State: 111000100010 - Reward: 1.50
State: 111000100011 - Reward: 1.68
State: 111000100100 - Reward: 0.57
State: 111000100101 - Reward: 1.69
State: 111000100110 - Reward: 1.62
State: 111000100111 - Reward: 1.05
State: 111000101000 - Reward: 0.05
State: 111000101001 - Reward: 0.29
State: 111000101010 - Reward: 1.34
State: 111000101011 - Reward: 0.40
State: 111000101100 - Reward: 1.50
State: 111000101101 - Reward: 0.32
State: 111000101110 - Reward: 0.57
State: 111000101111 - Reward: 0.50
State: 111000110000 - Reward: 1.68
State: 111000110001 - Reward: 1.38
State: 111000110010 - Reward: 0.59
State: 111000110011 - Reward: 1.51
State: 111000110100 - Reward: 0.06
State: 111000110101 - Reward: 1.63
State: 111000110110 - Reward: 0.20
State: 111000110111 - Reward: 1.74
State: 111000111000 - Reward: 1.48
State: 111000111001 - Reward: 1.73
State: 111000111010 - Reward: 1.48
State: 111000111011 - Reward: 1.12
State: 111000111100 - Reward: 0.48
State: 111000111101 - Reward: 1.57
State: 111000111110 - Reward: 1.60
State: 111000111111 - Reward: 0.58
State: 111001000000 - Reward: 1.33
State: 111001000001 - Reward: 1.85
State: 111001000010 - Reward: 0.78
State: 111001000011 - Reward: 1.91
State: 111001000100 - Reward: 1.95
State: 111001000101 - Reward: 0.63
State: 111001000110 - Reward: 1.10
State: 111001000111 - Reward: 0.03
State: 111001001000 - Reward: 0.50
State: 111001001001 - Reward: 1.24
State: 111001001010 - Reward: 1.56
State: 111001001011 - Reward: 1.74
State: 111001001100 - Reward: 1.66
State: 111001001101 - Reward: 1.82
State: 111001001110 - Reward: 1.41
State: 111001001111 - Reward: 1.30
```

```
State: 111001010000 - Reward: 1.51
State: 111001010001 - Reward: 1.09
State: 111001010010 - Reward: 1.21
State: 111001010011 - Reward: 1.55
State: 111001010100 - Reward: 1.93
State: 111001010101 - Reward: 0.59
State: 111001010110 - Reward: 0.35
State: 111001010111 - Reward: 1.37
State: 111001011000 - Reward: 0.37
State: 111001011001 - Reward: 0.35
State: 111001011010 - Reward: 1.03
State: 111001011011 - Reward: 0.75
State: 111001011100 - Reward: 0.86
State: 111001011101 - Reward: 1.11
State: 111001011110 - Reward: 0.26
State: 111001011111 - Reward: 1.17
State: 111001100000 - Reward: 0.51
State: 111001100001 - Reward: 0.66
State: 111001100010 - Reward: 1.42
State: 111001100011 - Reward: 0.31
State: 111001100100 - Reward: 0.31
State: 111001100101 - Reward: 0.65
State: 111001100110 - Reward: 0.10
State: 111001100111 - Reward: 1.86
State: 111001101000 - Reward: 1.23
State: 111001101001 - Reward: 1.32
State: 111001101010 - Reward: 0.98
State: 111001101011 - Reward: 1.15
State: 111001101100 - Reward: 0.72
State: 111001101101 - Reward: 1.57
State: 111001101110 - Reward: 0.64
State: 111001101111 - Reward: 0.44
State: 111001110000 - Reward: 0.37
State: 111001110001 - Reward: 0.14
State: 111001110010 - Reward: 1.01
State: 111001110011 - Reward: 0.83
State: 111001110100 - Reward: 1.07
State: 111001110101 - Reward: 0.18
State: 111001110110 - Reward: 0.44
State: 111001110111 - Reward: 0.43
State: 111001111000 - Reward: 0.66
State: 111001111001 - Reward: 0.72
State: 111001111010 - Reward: 0.44
State: 111001111011 - Reward: 1.51
State: 111001111100 - Reward: 1.06
State: 111001111101 - Reward: 1.99
State: 111001111110 - Reward: 1.65
State: 111001111111 - Reward: 1.96
State: 111010000000 - Reward: 0.02
State: 111010000001 - Reward: 1.34
State: 111010000010 - Reward: 0.89
State: 111010000011 - Reward: 1.81
```

```
State: 111010000100 - Reward: 1.23
State: 111010000101 - Reward: 1.24
State: 111010000110 - Reward: 1.92
State: 111010000111 - Reward: 1.37
State: 111010001000 - Reward: 0.64
State: 111010001001 - Reward: 1.83
State: 111010001010 - Reward: 1.89
State: 111010001011 - Reward: 0.77
State: 111010001100 - Reward: 1.08
State: 111010001101 - Reward: 0.57
State: 111010001110 - Reward: 1.82
State: 111010001111 - Reward: 1.64
State: 111010010000 - Reward: 0.75
State: 111010010001 - Reward: 1.61
State: 111010010010 - Reward: 0.89
State: 111010010011 - Reward: 0.09
State: 111010010100 - Reward: 1.80
State: 111010010101 - Reward: 0.39
State: 111010010110 - Reward: 1.03
State: 111010010111 - Reward: 1.90
State: 111010011000 - Reward: 0.33
State: 111010011001 - Reward: 1.91
State: 111010011010 - Reward: 1.08
State: 111010011011 - Reward: 0.01
State: 111010011100 - Reward: 0.13
State: 111010011101 - Reward: 1.34
State: 111010011110 - Reward: 1.55
State: 111010011111 - Reward: 1.73
State: 111010100000 - Reward: 0.85
State: 111010100001 - Reward: 0.21
State: 111010100010 - Reward: 1.08
State: 111010100011 - Reward: 1.41
State: 111010100100 - Reward: 1.95
State: 111010100101 - Reward: 1.55
State: 111010100110 - Reward: 1.29
State: 111010100111 - Reward: 1.88
State: 111010101000 - Reward: 1.49
State: 111010101001 - Reward: 0.31
State: 111010101010 - Reward: 0.92
State: 111010101011 - Reward: 0.66
State: 111010101100 - Reward: 0.18
State: 111010101101 - Reward: 0.11
State: 111010101110 - Reward: 1.59
State: 111010101111 - Reward: 1.12
State: 111010110000 - Reward: 1.15
State: 111010110001 - Reward: 0.46
State: 111010110010 - Reward: 0.52
State: 111010110011 - Reward: 0.78
State: 111010110100 - Reward: 1.26
State: 111010110101 - Reward: 0.87
State: 111010110110 - Reward: 0.03
State: 111010110111 - Reward: 1.35
```

```
State: 111010111000 - Reward: 1.07
State: 111010111001 - Reward: 1.28
State: 111010111010 - Reward: 1.24
State: 111010111011 - Reward: 1.51
State: 111010111100 - Reward: 1.17
State: 111010111101 - Reward: 1.40
State: 111010111110 - Reward: 0.15
State: 111010111111 - Reward: 1.86
State: 111011000000 - Reward: 0.21
State: 111011000001 - Reward: 1.57
State: 111011000010 - Reward: 0.60
State: 111011000011 - Reward: 0.17
State: 111011000100 - Reward: 1.53
State: 111011000101 - Reward: 0.88
State: 111011000110 - Reward: 0.79
State: 111011000111 - Reward: 1.32
State: 111011001000 - Reward: 0.95
State: 111011001001 - Reward: 1.07
State: 111011001010 - Reward: 0.27
State: 111011001011 - Reward: 0.78
State: 111011001100 - Reward: 1.59
State: 111011001101 - Reward: 1.09
State: 111011001110 - Reward: 1.92
State: 111011001111 - Reward: 0.29
State: 111011010000 - Reward: 1.35
State: 111011010001 - Reward: 1.83
State: 111011010010 - Reward: 1.59
State: 111011010011 - Reward: 1.46
State: 111011010100 - Reward: 0.75
State: 111011010101 - Reward: 1.90
State: 111011010110 - Reward: 1.11
State: 111011010111 - Reward: 1.11
State: 111011011000 - Reward: 0.25
State: 111011011001 - Reward: 0.01
State: 111011011010 - Reward: 1.19
State: 111011011011 - Reward: 1.07
State: 111011011100 - Reward: 1.89
State: 111011011101 - Reward: 0.61
State: 111011011110 - Reward: 1.50
State: 111011011111 - Reward: 1.81
State: 111011100000 - Reward: 0.69
State: 111011100001 - Reward: 0.83
State: 111011100010 - Reward: 1.29
State: 111011100011 - Reward: 1.03
State: 111011100100 - Reward: 0.32
State: 111011100101 - Reward: 0.44
State: 111011100110 - Reward: 1.67
State: 111011100111 - Reward: 0.39
State: 111011101000 - Reward: 0.36
State: 111011101001 - Reward: 1.60
State: 111011101010 - Reward: 1.70
State: 111011101011 - Reward: 1.70
```

```
State: 111011101100 - Reward: 1.86
State: 111011101101 - Reward: 1.99
State: 111011101110 - Reward: 0.92
State: 111011101111 - Reward: 1.10
State: 111011110000 - Reward: 0.58
State: 111011110001 - Reward: 0.13
State: 111011110010 - Reward: 0.20
State: 111011110011 - Reward: 1.45
State: 111011110100 - Reward: 0.97
State: 111011110101 - Reward: 0.66
State: 111011110110 - Reward: 0.26
State: 111011110111 - Reward: 1.31
State: 111011111000 - Reward: 0.20
State: 111011111001 - Reward: 1.24
State: 111011111010 - Reward: 1.80
State: 111011111011 - Reward: 0.64
State: 111011111100 - Reward: 0.90
State: 111011111101 - Reward: 1.23
State: 111011111110 - Reward: 0.61
State: 111011111111 - Reward: 1.17
State: 111100000000 - Reward: 1.13
State: 111100000001 - Reward: 0.73
State: 111100000010 - Reward: 0.63
State: 111100000011 - Reward: 0.86
State: 111100000100 - Reward: 0.01
State: 111100000101 - Reward: 0.49
State: 111100000110 - Reward: 0.44
State: 111100000111 - Reward: 1.48
State: 111100001000 - Reward: 0.87
State: 111100001001 - Reward: 1.68
State: 111100001010 - Reward: 0.27
State: 111100001011 - Reward: 1.47
State: 111100001100 - Reward: 1.76
State: 111100001101 - Reward: 0.93
State: 111100001110 - Reward: 0.72
State: 111100001111 - Reward: 0.61
State: 111100010000 - Reward: 1.10
State: 111100010001 - Reward: 0.35
State: 111100010010 - Reward: 1.21
State: 111100010011 - Reward: 1.68
State: 111100010100 - Reward: 1.72
State: 111100010101 - Reward: 0.28
State: 111100010110 - Reward: 1.08
State: 111100010111 - Reward: 0.53
State: 111100011000 - Reward: 1.77
State: 111100011001 - Reward: 0.15
State: 111100011010 - Reward: 0.15
State: 111100011011 - Reward: 0.04
State: 111100011100 - Reward: 1.01
State: 111100011101 - Reward: 0.06
State: 111100011110 - Reward: 1.16
State: 111100011111 - Reward: 0.81
```

```
State: 111100100000 - Reward: 1.18
State: 111100100001 - Reward: 1.81
State: 111100100010 - Reward: 1.10
State: 111100100011 - Reward: 1.09
State: 111100100100 - Reward: 2.00
State: 111100100101 - Reward: 0.94
State: 111100100110 - Reward: 1.55
State: 111100100111 - Reward: 0.73
State: 111100101000 - Reward: 0.45
State: 111100101001 - Reward: 1.54
State: 111100101010 - Reward: 1.47
State: 111100101011 - Reward: 0.58
State: 111100101100 - Reward: 0.93
State: 111100101101 - Reward: 1.02
State: 111100101110 - Reward: 0.79
State: 111100101111 - Reward: 1.00
State: 111100110000 - Reward: 1.33
State: 111100110001 - Reward: 1.70
State: 111100110010 - Reward: 1.61
State: 111100110011 - Reward: 1.23
State: 111100110100 - Reward: 0.33
State: 111100110101 - Reward: 1.03
State: 111100110110 - Reward: 0.89
State: 111100110111 - Reward: 0.36
State: 111100111000 - Reward: 1.90
State: 111100111001 - Reward: 1.32
State: 111100111010 - Reward: 1.94
State: 111100111011 - Reward: 1.47
State: 111100111100 - Reward: 0.97
State: 111100111101 - Reward: 0.72
State: 111100111110 - Reward: 0.44
State: 111100111111 - Reward: 0.98
State: 111101000000 - Reward: 0.13
State: 111101000001 - Reward: 0.74
State: 111101000010 - Reward: 0.09
State: 111101000011 - Reward: 0.41
State: 111101000100 - Reward: 1.82
State: 111101000101 - Reward: 0.72
State: 111101000110 - Reward: 0.94
State: 111101000111 - Reward: 0.91
State: 111101001000 - Reward: 0.09
State: 111101001001 - Reward: 1.96
State: 111101001010 - Reward: 0.65
State: 111101001011 - Reward: 1.41
State: 111101001100 - Reward: 1.04
State: 111101001101 - Reward: 1.66
State: 111101001110 - Reward: 1.67
State: 111101001111 - Reward: 0.53
State: 111101010000 - Reward: 1.09
State: 111101010001 - Reward: 0.35
State: 111101010010 - Reward: 1.31
State: 111101010011 - Reward: 0.73
```

```
State: 111101010100 - Reward: 1.31
State: 111101010101 - Reward: 1.67
State: 111101010110 - Reward: 1.03
State: 111101010111 - Reward: 0.75
State: 111101011000 - Reward: 1.81
State: 111101011001 - Reward: 1.03
State: 111101011010 - Reward: 0.71
State: 111101011011 - Reward: 1.74
State: 111101011100 - Reward: 0.97
State: 111101011101 - Reward: 0.96
State: 111101011110 - Reward: 1.21
State: 111101011111 - Reward: 1.00
State: 111101100000 - Reward: 0.28
State: 111101100001 - Reward: 0.33
State: 111101100010 - Reward: 0.15
State: 111101100011 - Reward: 1.29
State: 111101100100 - Reward: 0.42
State: 111101100101 - Reward: 0.37
State: 111101100110 - Reward: 0.72
State: 111101100111 - Reward: 1.43
State: 111101101000 - Reward: 0.24
State: 111101101001 - Reward: 0.46
State: 111101101010 - Reward: 1.62
State: 111101101011 - Reward: 1.44
State: 111101101100 - Reward: 0.96
State: 111101101101 - Reward: 0.96
State: 111101101110 - Reward: 0.42
State: 111101101111 - Reward: 0.32
State: 111101110000 - Reward: 1.67
State: 111101110001 - Reward: 0.04
State: 111101110010 - Reward: 0.09
State: 111101110011 - Reward: 1.15
State: 111101110100 - Reward: 0.32
State: 111101110101 - Reward: 1.26
State: 111101110110 - Reward: 0.08
State: 111101110111 - Reward: 1.14
State: 111101111000 - Reward: 0.11
State: 111101111001 - Reward: 0.52
State: 111101111010 - Reward: 0.36
State: 111101111011 - Reward: 1.92
State: 111101111100 - Reward: 1.20
State: 111101111101 - Reward: 1.13
State: 111101111110 - Reward: 0.04
State: 111101111111 - Reward: 1.44
State: 111110000000 - Reward: 1.32
State: 111110000001 - Reward: 0.57
State: 111110000010 - Reward: 0.17
State: 111110000011 - Reward: 0.90
State: 111110000100 - Reward: 1.99
State: 111110000101 - Reward: 1.73
State: 111110000110 - Reward: 0.34
State: 111110000111 - Reward: 1.66
```

```
State: 111110001000 - Reward: 1.20
State: 111110001001 - Reward: 1.59
State: 111110001010 - Reward: 1.64
State: 111110001011 - Reward: 0.36
State: 111110001100 - Reward: 1.32
State: 111110001101 - Reward: 0.53
State: 111110001110 - Reward: 1.45
State: 111110001111 - Reward: 0.69
State: 111110010000 - Reward: 0.91
State: 111110010001 - Reward: 1.18
State: 111110010010 - Reward: 0.46
State: 111110010011 - Reward: 0.77
State: 111110010100 - Reward: 0.22
State: 111110010101 - Reward: 0.40
State: 111110010110 - Reward: 1.72
State: 111110010111 - Reward: 1.01
State: 111110011000 - Reward: 0.84
State: 111110011001 - Reward: 0.30
State: 111110011010 - Reward: 0.19
State: 111110011011 - Reward: 0.95
State: 111110011100 - Reward: 1.23
State: 111110011101 - Reward: 0.08
State: 111110011110 - Reward: 1.57
State: 111110011111 - Reward: 1.01
State: 111110100000 - Reward: 0.24
State: 111110100001 - Reward: 0.96
State: 111110100010 - Reward: 0.26
State: 111110100011 - Reward: 1.21
State: 111110100100 - Reward: 1.66
State: 111110100101 - Reward: 1.79
State: 111110100110 - Reward: 1.55
State: 111110100111 - Reward: 1.30
State: 111110101000 - Reward: 1.04
State: 111110101001 - Reward: 0.74
State: 111110101010 - Reward: 0.09
State: 111110101011 - Reward: 1.06
State: 111110101100 - Reward: 0.46
State: 111110101101 - Reward: 1.60
State: 111110101110 - Reward: 1.58
State: 111110101111 - Reward: 0.76
State: 111110110000 - Reward: 1.18
State: 111110110001 - Reward: 1.48
State: 111110110010 - Reward: 1.52
State: 111110110011 - Reward: 1.39
State: 111110110100 - Reward: 0.20
State: 111110110101 - Reward: 0.27
State: 111110110110 - Reward: 0.95
State: 111110110111 - Reward: 0.47
State: 111110111000 - Reward: 1.72
State: 111110111001 - Reward: 0.57
State: 111110111010 - Reward: 1.75
State: 111110111011 - Reward: 0.82
```

```
State: 111110111100 - Reward: 0.38
State: 111110111101 - Reward: 1.42
State: 111110111110 - Reward: 1.58
State: 111110111111 - Reward: 1.16
State: 111111000000 - Reward: 0.24
State: 111111000001 - Reward: 0.01
State: 111111000010 - Reward: 1.31
State: 111111000011 - Reward: 1.38
State: 111111000100 - Reward: 0.63
State: 111111000101 - Reward: 0.72
State: 111111000110 - Reward: 0.31
State: 111111000111 - Reward: 1.30
State: 111111001000 - Reward: 0.51
State: 111111001001 - Reward: 1.71
State: 111111001010 - Reward: 0.85
State: 111111001011 - Reward: 0.73
State: 111111001100 - Reward: 0.55
State: 111111001101 - Reward: 1.36
State: 111111001110 - Reward: 1.51
State: 111111001111 - Reward: 0.83
State: 111111010000 - Reward: 1.57
State: 111111010001 - Reward: 0.96
State: 111111010010 - Reward: 0.74
State: 111111010011 - Reward: 1.11
State: 111111010100 - Reward: 0.51
State: 111111010101 - Reward: 0.61
State: 111111010110 - Reward: 0.69
State: 111111010111 - Reward: 1.41
State: 111111011000 - Reward: 1.47
State: 111111011001 - Reward: 1.71
State: 111111011010 - Reward: 1.32
State: 111111011011 - Reward: 1.50
State: 111111011100 - Reward: 0.89
State: 111111011101 - Reward: 1.40
State: 111111011110 - Reward: 0.32
State: 111111011111 - Reward: 0.43
State: 111111100000 - Reward: 0.80
State: 111111100001 - Reward: 0.68
State: 111111100010 - Reward: 1.10
State: 111111100011 - Reward: 1.39
State: 111111100100 - Reward: 1.41
State: 111111100101 - Reward: 0.32
State: 111111100110 - Reward: 1.93
State: 111111100111 - Reward: 0.01
State: 111111101000 - Reward: 0.18
State: 111111101001 - Reward: 0.29
State: 111111101010 - Reward: 1.85
State: 111111101011 - Reward: 0.87
State: 111111101100 - Reward: 0.13
State: 111111101101 - Reward: 0.44
State: 111111101110 - Reward: 0.16
State: 111111101111 - Reward: 0.08
```

```
State: 111111110000 - Reward: 0.77
State: 111111110001 - Reward: 1.97
State: 111111110010 - Reward: 1.23
State: 111111110011 - Reward: 1.04
State: 111111110100 - Reward: 1.42
State: 111111110101 - Reward: 1.19
State: 111111110110 - Reward: 1.89
State: 111111110111 - Reward: 1.64
State: 111111111000 - Reward: 1.28
State: 111111111001 - Reward: 0.88
State: 111111111010 - Reward: 0.40
State: 111111111011 - Reward: 1.31
State: 111111111100 - Reward: 1.61
State: 111111111101 - Reward: 0.57
State: 111111111110 - Reward: 0.07
State: 111111111111 - Reward: 1.20

True Optimal Path: root -> 0 -> 00 -> 001 -> 0010 -> 00100 -> 001000 ->
0010000 -> 00100000 -> 001000001 -> 0010000010 -> 00100000100 -> 001000
001000
True Optimal Reward: 2.500
```

In [ ]:

In [ ]: